Supplementary information for:

Hetero-interpenetrated metal-organic frameworks

David Perl, Seok. J. Lee, Alan Ferguson, Geoffrey B. Jameson and Shane G. Telfer

MacDiarmid Institute for Advanced Materials and *Nanotechnology,*

School of Natural Sciences, Massey University, Palmerston North, New Zealand.

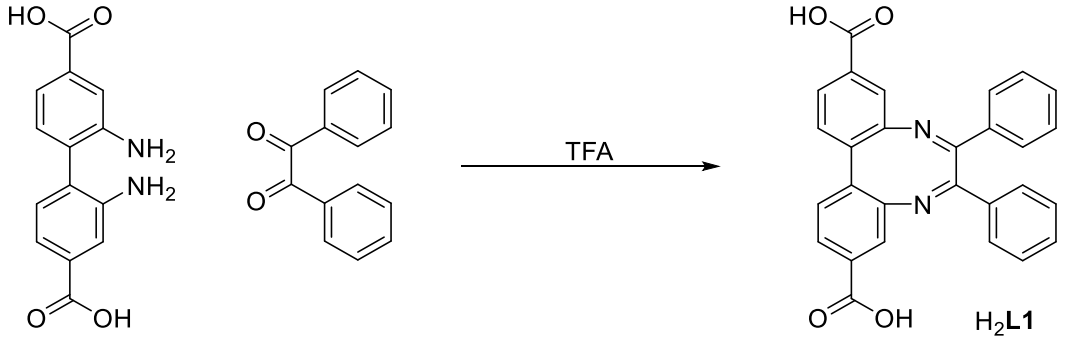
Email: s.telfer@massey.ac.nz

Contents:

S1. General Procedures

All starting materials and solvents were used as received from commercial sources without further purification unless otherwise noted. Column chromatography was carried out on silica gel (grade 60, mesh size 230-400, Scharlau). NMR spectra were recorded at room temperature (unless otherwise noted) on Bruker-400 and Bruker-500 Avance instruments, with the use of the solvent proton as an internal standard. High performance liquid chromatography (HPLC) was carried out using a Dionex Ultimate 3000 system equipped with a UV detector. Elemental analyses were performed by the Campbell Microanalytical Laboratory at the University of Otago, New Zealand. PXRD patterns were obtained on a Rigaku Spider equipped with a copper rotating anode generator (λ = 1.54 Å). Atomic Absorption Spectroscopy was performed using a GBC XplorAA. SEM images and optical micrographs were collected at the MMIC, Massey University, Palmerston North, New Zealand using a FEI Quanta 200 Environmental Scanning Electron Microscope (SEM) with EDAX module and a Zeiss Axiophot Microscope with Differential Interference Contrast (DIC) Optics and Colour CCD camera, respectively.

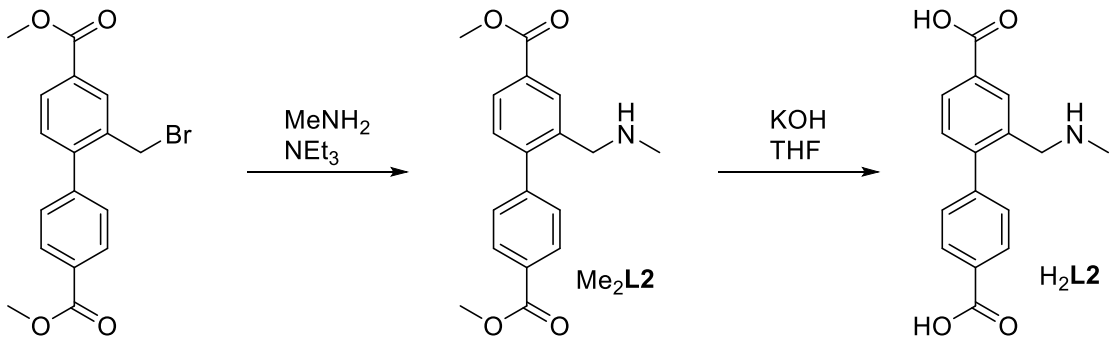
S1.1. Synthesis of H_2L1



Scheme S1: Synthesis of H_2**L1**.

2,2´-Diaminobiphenyl-4,4´-dicarboxylic acid (300mg, 1.10 mmol) and benzil (300 mg, 1.43 mmol) in dioxane (7.2 mL) and TFA (0.8 mL) were heated to 200 °C by microwave irradiation for 2 hours. The reaction mixture was transferred to a centrifuge tube and water (25 mL) was added to precipitate the product. The solid was collected by centrifugation at 4000 rpm for 15 minutes. The crude product was dissolved in 1 M aqueous NaOH (10 mL) and washed with CH_2Cl_2 (5 mL × 3). Aqueous HCl (2 M) was then added dropwise until a pale-yellow precipitate formed, which was collected by filtration and washed with H_2O to yield H_2**L1** (420 mg, 1.10 mmol, 85%). Characterisation data matched those previously reported.[1, 2]

S1.2. Synthesis of H_2L2



Scheme S2: Synthesis of H_2**L2**.

Dimethyl-2-(bromomethyl)-4,4'-biphenyldicarboxylate[3] (500 mg, 1.38 mmol) was dissolved in Et_2O (30 mL) in a 100 mL round-bottom flask equipped with a magnetic stirrer. 1 M $MeNH_2$ in Et_2O (20 mL) and Et_3N (1 mL) were added and the mixture stirred at room temperature overnight. The reaction mixture was washed with H_2O (20 mL × 3) and concentrated under reduced pressure, then the residue chromatographed on silica in a gradient from 1:0.01 CH_2Cl_2:MeOH to 1:0.2 CH_2Cl_2:MeOH to yield dimethyl 2-(methylaminomethyl)-4,4'-biphenyldicarboxylate (Me_2**L2**, 382 mg, 1.21 mmol, 88%).

^{1}H NMR (500 MHz, $CDCl_3$): 8.20 (1H, s), 8.12 (2H, d, J = 8.17 Hz), 8.00 (1H, dd, J = 7.98, 1.50 Hz), 7.48 (2H, d, J = 8.17 Hz), 7.33 (1H, d, J = 8.00 Hz), 3.97 (3H, s), 3.95 (3H, s), 3.72 (2H, s), 2.36 (2H, s).

^{13}C NMR (125 MHz, $CDCl_3$): 166.82, 145.46, 144.82, 130.47, 130.09, 129.89, 129.66, 129.52, 129.04, 128.41, 52.25, 52.23, 35.63

ESI-MS (-): Calc. 313.35 found 313.08 (M^-)

Dimethyl 2-(methylaminomethyl)-4,4'-biphenyldicarboxylate (Me_2**L2**, 300 mg, 0.96 mmol) was dissolved in THF (15 mL) in a 50 mL round-bottom flask equipped with a stir bar. 2 M aqueous KOH (4 mL) was added and the mixture stirred vigorously overnight at room temperature. THF was removed under reduced pressure, and the mixture carefully neutralized with 1 M aqueous HCl, upon which a white precipitate formed. The precipitate of H_2**L2** was collected by filtration, washed with H_2O (5 mL × 3) and dried under high vacuum (256 mg, 0.90 mmol, 94%).

^{1}H NMR (500 MHz, d_6-DMSO): 10.90 (2H, s, b), 8.36 (1H, s), 8.07 (2H, d, J = 7.25 Hz), 8.03 (1H, d, J = 7.75 Hz), 7.57 (2H, d, J = 7.15 Hz), 7.49 (1H, d, J = 7.50 Hz), 4.09 (2H, s), 2.47 (3H, s).

^{13}C NMR (125 MHz, d_6-DMSO): 167.42, 146.90, 145.6, 131.26, 130.88, 130.77, 130.47, 130.11, 13.03 130.02, 49.10, 33.43.

ESI-MS (+): Calc. 286.11 found 286.11 (MH^+), (-): Calc 284.09 found 284.09 ($M - H^+$).

S1.3. MOF digestion

MOF samples were washed with DMF three times, then acetone five times. Excess solvent was removed, and the MOF dried under vacuum overnight. To the dried MOF, 0.2 mL of a 375 mM solution of DCl in d_6-DMSO was first added, and the mixture sonicated until the MOF was dissolved, then a further 0.4 mL of d_6-DMSO added. This solution was then used for ^{1}H NMR spectroscopy before being further used for AAS where relevant.

S1.4. Atomic Absorption Spectroscopy

Digested MOF samples were diluted to 10 mL in 2 M aqueous HCl, then filtered through a syringe filter and analysed using a GBC XplorAA for their cobalt and zinc contents. The absolute wt% of these metals were not determined, instead, only their ratio was obtained.

S2. Synthesis, crystallographic details, and characterisation of MUFs-91 – 93

S2.1. General crystallographic methods and strategy

Powder X-ray diffraction data and some single crystal X-ray diffraction data were obtained using a Rigaku Spider diffractometer using Cu_α radiation. Other single crystal X-ray diffraction data were obtained on the MX1 and MX2 beamlines at the Australian Synchrotron, part of ANSTO.[4, 5] In no cases were any solvent scattering contributions been taken into account. The crystallographic models used were adapted from that for MUF-9,[1] which was originally solved with SHELXT[6], and SHELXL[7] was used for refinement using OLEX2 as a graphical interface and for the preparation of publication material.[8]

A crystallographic model for MUF-91 was developed from the highest-quality data set (which had a high percentage of interpenetration, PIP%). The coordinates of the atoms were then fixed, with the occupancy of the secondary lattice assigned to a free variable. This model was then used as a starting point for the refinement of the datasets obtained from the lower PIP% samples (for which the data quality was lower). Tight restraints were added to preserve the refinement stability and chemical correctness. Datasets were refined individually using this restrained model, with the restraints loosened as far as the quality of each dataset would allow. The cell parameters were determined individually for each dataset.

A similar model was used to refine the MUF-92 and MUF-93 datasets, only making modifications according to the identity of the secondary (interpenetrating) lattice. For example, for MUF-92, the same model as for MUF-91 was used, except an amino group was added in the appropriate location and with the appropriate disorder parameters.

S2.1.1. A note about data quality and CheckCIF alerts

Structural models were determined from high-quality datasets and refined appropriately, resulting in good statistics – these are referred to in the following tables crystallographic results as the ‘best’ data. Many of the other datasets have incurred one or more level-A CheckCIF alerts, notably those relating to low resolution (e.g. ‘The value of sine(theta_max)/wavelength is less than 0.550’), data/parameter ratios, and high R_{int} or wR_2 values. It is necessary to use the datasets that give rise to these alerts because they provide important information about the level of interpenetration; relying only on good datasets would bias the results towards higher values of PIP% among other issues. Also, with atom positions fixed, these datasets contain sufficient information to determine the value of PIP% to reasonable accuracy and precision, and they are not used for structure determination *per se*.

S2.1.2. Determination of PIP% by PXRD

Baselines were removed from PXRD data using a customised Sonneveld-Visser algorithm (python code in S9.1), and patterns scaled to an arbitrary intensity of 1000. The apparent PIP% was obtained by calculating the contribution of each component in a linear combination of a fully interpenetrated and a non-interpenetrated sample of MUF-9, with the least-squares difference to the observed PXRD pattern (python code in S9.2).

S2.1.3. Summary of CIF files uploaded as supplementary information

Table S1: Summary of the CIF files uploaded as supplementary information.

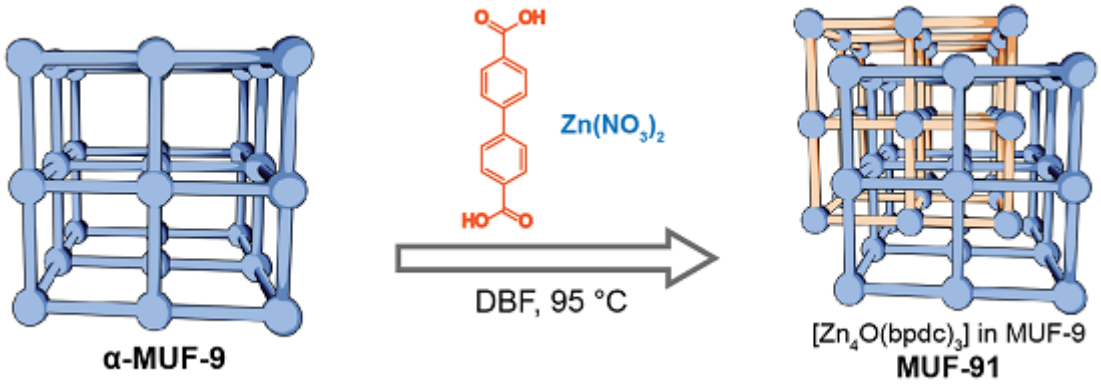
File name	MOF	CCDC number	Formula	Notes
MUF-91-1h-12pc.cif	MUF-91	2144464	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.12}$	PIP = 12%
MUF-91-3h-23pc.cif	MUF-91	2144465	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.23}$	PIP = 23%
MUF-91-3h-27pc.cif	MUF-91	2144466	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.27}$	PIP = 27%
MUF-91-3h-38pc.cif	MUF-91	2144467	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.38}$	PIP = 38%
MUF-91-3h-41pc.cif	MUF-91	2144468	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.41}$	PIP = 41%
MUF-91-6h-50pc.cif	MUF-91	2144469	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.50}$	PIP = 50%
MUF-91-6h-52pc.cif	MUF-91	2144470	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.52}$	PIP = 52%
MUF-91-9h-51pc.cif	MUF-91	2144471	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.51}$	PIP = 51%
MUF-91-9h-64pc.cif	MUF-91	2144472	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.64}$	PIP = 64%
MUF-91-9h-76pc.cif	MUF-91	2144473	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.76}$	PIP = 76%
MUF-91-12h-57pc.cif	MUF-91	2144458	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.57}$	PIP = 57%
MUF-91-12h-71pc.cif	MUF-91	2144459	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.71}$	PIP = 71%
MUF-91-15h-51pc.cif	MUF-91	2144460	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.51}$	PIP = 51%, shell present
MUF-91-15h-58pc.cif	MUF-91	2144461	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.58}$	PIP = 58%, shell present
MUF-91-15h-68pc.cif	MUF-91	2144462	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.68}$	PIP = 68%, shell present
MUF-91-18h-46pc.cif	MUF-91	2144463	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc)_3]_{0.46}$	PIP = 46%, shell present
MUF-92-3h-42pc.cif	MUF-92	2144580	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.41}$	PIP = 42%
MUF-92-3h-48pc.cif	MUF-92	2144581	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.47}$	PIP = 48%
MUF-92-3h-50pc.cif	MUF-92	2144582	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.50}$	PIP = 50%
MUF-92-6h-53pc.cif	MUF-92	2144583	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.53}$	PIP = 53%
MUF-92-6h-63pc.cif	MUF-92	2144584	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.63}$	PIP = 63%
MUF-92-6h-65pc.cif	MUF-92	2144585	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.65}$	PIP = 65%
MUF-92-9h-63pc.cif	MUF-92	2144586	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.63}$	PIP = 63%
MUF-92-9h-65pc.cif	MUF-92	2144587	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.65}$	PIP = 65%
MUF-92-9h-71pc.cif	MUF-92	2144588	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.71}$	PIP = 71%
MUF-92-12h-68pc.cif	MUF-92	2144574	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.68}$	PIP = 68%, shell present
MUF-92-12h-71pc.cif	MUF-92	2144575	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.71}$	PIP = 71%, shell present
MUF-92-12h-81pc.cif	MUF-92	2144576	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.81}$	PIP = 81%, shell present
MUF-92-15h-71pc.cif	MUF-92	2144577	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.71}$	PIP = 71%, shell present
MUF-92-15h-77pc-2.cif	MUF-92	2144578	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.77}$	PIP = 77%, shell present

MUF-92-15h-77pc.cif	MUF-92	2144579	$[Zn_4O(\mathbf{L1})_3][Zn_4O(bpdc\text{-}NH_2)_3]_{0.77}$	PIP = 77%, shell present
MUF-93-12h-01pc.cif	MUF-93	2149445	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.01}$	PIP = 1%
MUF-93-12h-02pc.cif	MUF-93	2149446	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.02}$	PIP = 2%
MUF-93-12h-10pc.cif	MUF-93	2149447	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.10}$	PIP = 10%
MUF-93-12h-4pc.cif	MUF-93	2149448	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.04}$	PIP = 4%
MUF-93-24h-14pc.cif	MUF-93	2149449	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.14}$	PIP = 14%
MUF-93-24h-15pc.cif	MUF-93	2149450	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.15}$	PIP = 15%
MUF-93-24h-16pc.cif	MUF-93	2149451	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.16}$	PIP = 16%
MUF-93-24h-21pc.cif	MUF-93	2149452	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.21}$	PIP = 21%
MUF-93-24h-28pc.cif	MUF-93	2149453	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.28}$	PIP = 28%
MUF-93-36h-22pc.cif	MUF-93	2149454	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.22}$	PIP = 22%
MUF-93-36h-28pc.cif	MUF-93	2149455	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.28}$	PIP = 28%
MUF-93-48h-39pc.cif	MUF-93	2149456	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.39}$	PIP = 39%
MUF-93-48h-48pc.cif	MUF-93	2149457	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.48}$	PIP = 48%
MUF-93-48h-51pc.cif	MUF-93	2149458	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.51}$	PIP = 51%
MUF-93-48h-61pc.cif	MUF-93	2149459	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.61}$	PIP = 61%
MUF-93-60h-59pc.cif	MUF-93	2149460	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.59}$	PIP = 59%
MUF-93-60h-63pc.cif	MUF-93	2149461	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.63}$	PIP = 63%
MUF-93-60h-73pc.cif	MUF-93	2149462	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.73}$	PIP = 73%
MUF-93-72h-62pc.cif	MUF-93	2149463	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.62}$	PIP = 62%, metal and L exchange
MUF-93-72h-63pc.cif	MUF-93	2149464	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.63}$	PIP = 63%, metal and L exchange
MUF-93-72h-76pc.cif	MUF-93	2149465	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.76}$	PIP = 76%, metal and L exchange
MUF-93-84h-66pc.cif	MUF-93	2149466	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.66}$	PIP = 66%, metal and L exchange
MUF-93-84h-67pc.cif	MUF-93	2149467	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.67}$	PIP = 67%, metal and L exchange
MUF-93-84h-71pc.cif	MUF-93	2149468	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.71}$	PIP = 71%, metal and L exchange
MUF-93-96h-66pc.cif	MUF-93	2149469	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.66}$	PIP = 66%, metal and L exchange
MUF-93-96h-67pc.cif	MUF-93	2149470	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.67}$	PIP = 67%, metal and L exchange
MUF-93-96h-70pc.cif	MUF-93	2149471	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.70}$	PIP = 70%, metal and L exchange
MUF-93-scan1-pos1-72pc.cif	MUF-93	2149430	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.72}$	Rastering
MUF-93-scan1-pos3-59pc.cif	MUF-93	2149431	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.59}$	Rastering
MUF-93-scan1-pos4-52pc.cif	MUF-93	2149432	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.52}$	Rastering
MUF-93-scan1-pos5-31pc.cif	MUF-93	2149433	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.31}$	Rastering
MUF-93-scan1-pos6-23pc.cif	MUF-93	2149434	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.23}$	Rastering
MUF-93-scan1-pos7-21pc.cif	MUF-93	2149435	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.21}$	Rastering
MUF-93-scan1-pos8-23pc.cif	MUF-93	2149436	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.23}$	Rastering

MUF-93-scan1-pos9-26pc.cif	MUF-93	2149437	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.26}$	Rastering.
MUF-93-scan1-pos10-31pc.cif	MUF-93	2149424	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.31}$	Rastering
MUF-93-scan1-pos11-47pc.cif	MUF-93	2149425	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.47}$	Rastering
MUF-93-scan1-pos12-59pc.cif	MUF-93	2149426	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.59}$	Rastering
MUF-93-scan1-pos13-65pc.cif	MUF-93	2149427	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.65}$	Rastering
MUF-93-scan1-pos14-73pc.cif	MUF-93	2149428	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.73}$	Rastering
MUF-93-scan1-pos15-73pc.cif	MUF-93	2149429	$[Zn_4O(\mathbf{L1})_3][Co_4O(bpdc)_3]_{0.73}$	Rastering

S2.2. MUF-91 (α-MUF-9 interpenetrated by $[Zn_4O(bpdc)_3]$)

S2.2.1. Synthesis of MUF-91 single crystals



α-MUF-9 was synthesised in a 4 mL glass vial with phenolic cap and PTFE-lined PDMS septum by a literature method.[1] This method uses 20 µmol of **L1** and yields 5-8 mg of α-MUF-9. A stock solution of H_2BPDC (0.5 mg mL^{-1}), $Zn(NO_3)_2 \cdot 4H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{-1}) was prepared in DBF. The solvent was removed from the α-MUF-9 crystals and replaced with this DBF stock solution, then the mixture was heated in a dry bath at 95 °C. The stock solution was removed and replaced with fresh solution every three hours. After the desired growth time, the crystals were removed from the dry bath, cooled to room temperature, and washed several times with DBF.

S2.2.2. PXRD

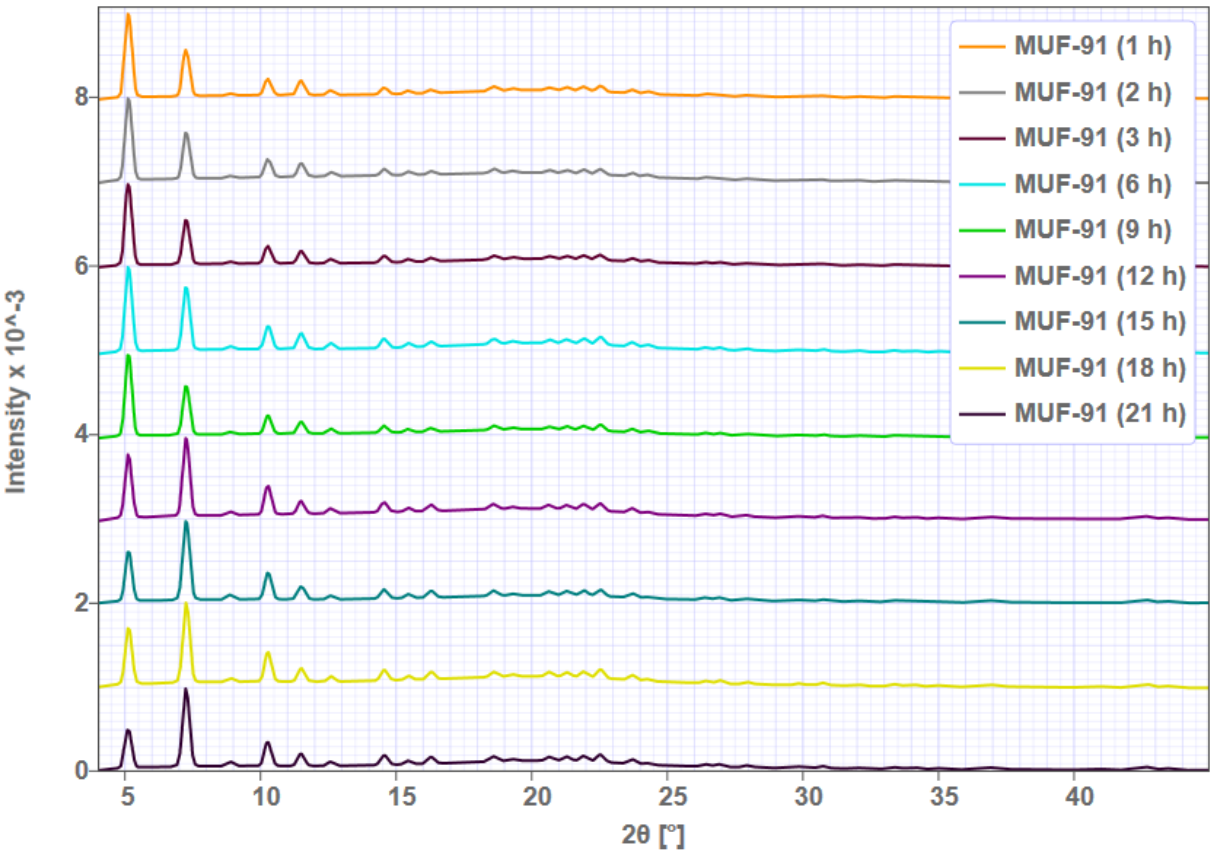


Figure S1: PXRD diffractograms of MUF-91 at various stages of growth.

S2.2.3. Crystallographic results

MUF-91 crystals grown over short reaction times have low PIP% and diffract poorly by nature, as observed for MUF-9 and MUF-10, due to their inhomogeneity. However, at higher PIP%, high-quality datasets were obtained (R_1 < 0.15 with no corrections for solvent scattering and resolution greater than 0.85 Å).

Table S2: Crystallographic results for selected MUF-91 datasets.

	Low PIP% (worst data)	**Medium PIP% (representative data)**	**High PIP% (best data)**
Identification code	**MUF-91-1h-12pc**	**MUF-91-3h-41pc**	**MUF-91-9h-64pc**
Growth time	1 hour	3 hours	9 hours
Empirical formula	$C_{89.23}H_{50.99}N_6O_{14.62}Zn_{4.50}$	$C_{101.41}H_{63.90}N_6O_{18.39}Zn_{5.66}$	$C_{110.85}H_{63.34}N_6O_{21.31}Zn_{6.56}$
Interpenetration fraction (PIP%)	0.12	0.41	0.64
Formula weight	1735.19	2028.54	2249.00
Temperature / K	100(2)	100(2)	100(2)
Crystal system		cubic	
Space group		P-43m	
a,b,c / Å	17.036(12)	17.119(2)	17.1300(12)
α, β, γ / °		90	
Volume / Å^3	4944(10)	5016.9(18)	5026.6(11)
Z		1	
ρ_{calc} / g cm^{-3}	0.583	0.671	0.743
μ / mm^{-1}	0.567	0.698	0.807
F(000) / e$^-$	880	1030	1138
Radiation		Synchrotron (λ = 0.71075 Å)	
2Θ range for data collection / °	5.348 to 29.402	5.322 to 41.632	5.318 to 46.496
Index ranges	-12 ≤ h ≤ 12, -10 ≤ k ≤ 12, -11 ≤ l ≤ 7	-17 ≤ h ≤ 17, -13 ≤ k ≤ 17, -17 ≤ l ≤ 13	-15 ≤ h ≤ 19, -19 ≤ k ≤ 19, -19 ≤ l ≤ 13
Reflections collected	3170	9849	13590
Independent reflections	395 [R_{int} = 0.1314, R_{sigma} = 0.0587]	1046 [R_{int} = 0.0759, R_{sigma} = 0.0305]	1409 [R_{int} = 0.0512, R_{sigma} = 0.0237]
Data/restraints/parameters	395/81/33	1046/177/99	1409/75/33
Goodness-of-fit on F^2	1.627	2.253	1.420
Final R indexes [I>=2σ (I)]	R_1 = 0.1956, wR_2 = 0.4522	R_1 = 0.1757, wR_2 = 0.4609	R_1 = 0.1305, wR_2 = 0.3263
Final R indexes [all data]	R_1 = 0.2072, wR_2 = 0.4628	R_1 = 0.1866, wR_2 = 0.4704	R_1 = 0.1498, wR_2 = 0.3557
Largest diff. peak/hole / e Å^{-3}	0.50/-0.51	1.87/-0.94	0.95/-0.38
Flack parameter	0.31(9)	0.620(18)	0.34(2)

S2.2.4. ^{1}H NMR spectroscopy

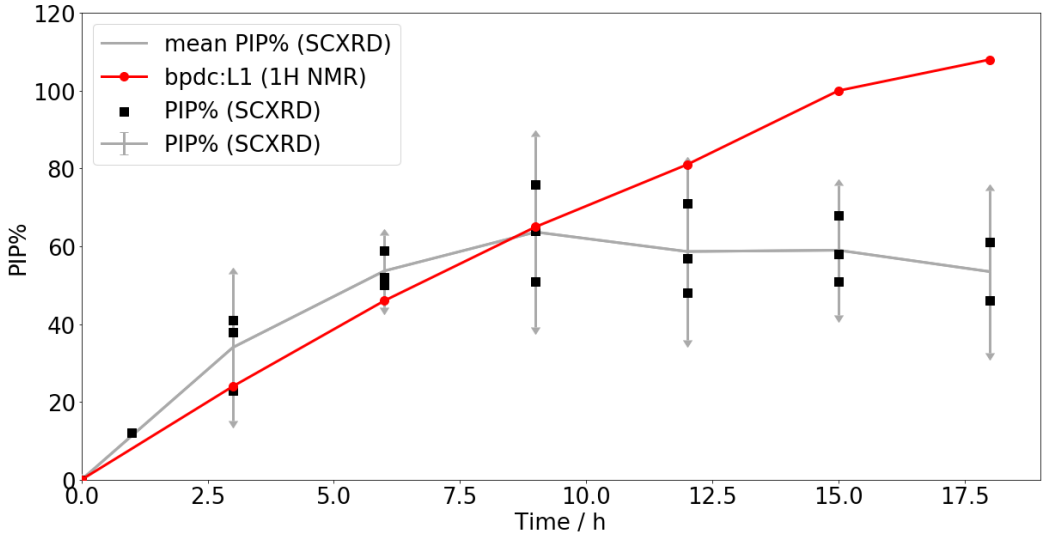


Figure S2: Plot of complementary characterisation data for MUF-91. Black squares represent the PIP% values obtained from SCXRD datasets collected at various time points in the growth of MUF-93. Red circles represent the ratio of BPDC to **L1** as determined by ^{1}H NMR spectroscopy. The grey line is drawn through the mean PIP% (from all SCXRD datasets) and the error bars correspond to the 95% confidence interval for the mean PIP% value for the sample.

S2.2.5. Optical Microscopy

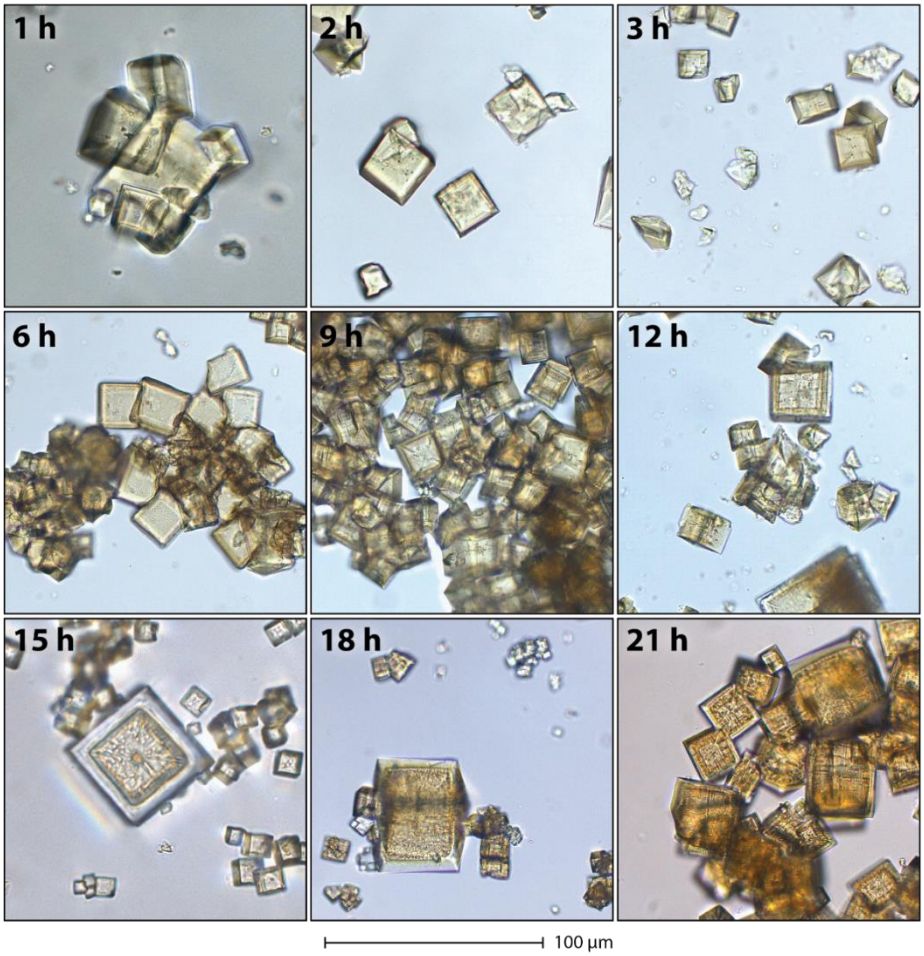


Figure S3. Optical micrographs of MUF-91 crystals after various growth times.

S2.2.6. Synthesis of microcrystalline MUF-91

S2.2.6.1. *Synthesis*

H_2**L1** (36 mg) and $Zn(NO_3)_2 \cdot 4H_2O$ (64 mg) were dissolved in DBF (4 mL) with H_2O (10 µL) in a 25 mL Schott bottle and heated in an 85°C oven for 6 hours, shaking the mixture every hour. The resulting small crystallites were then transferred to a 4 mL vial which was centrifuged at 1000 rpm for 30 seconds. The supernatant was replaced with 2 mL of a stock solution of H_2BPDC (0.5 mg mL^{-1}), $Zn(NO_3)_2 \cdot 4H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{-1}) in DBF. The mixture was then heated in a dry bath at 95 °C. At intervals of two hours, the mixture was centrifuged, a sample of crystallites taken out for analysis, the supernatant discarded and replaced with fresh stock solution, then the heating continued.

S2.2.6.2. *PXRD*

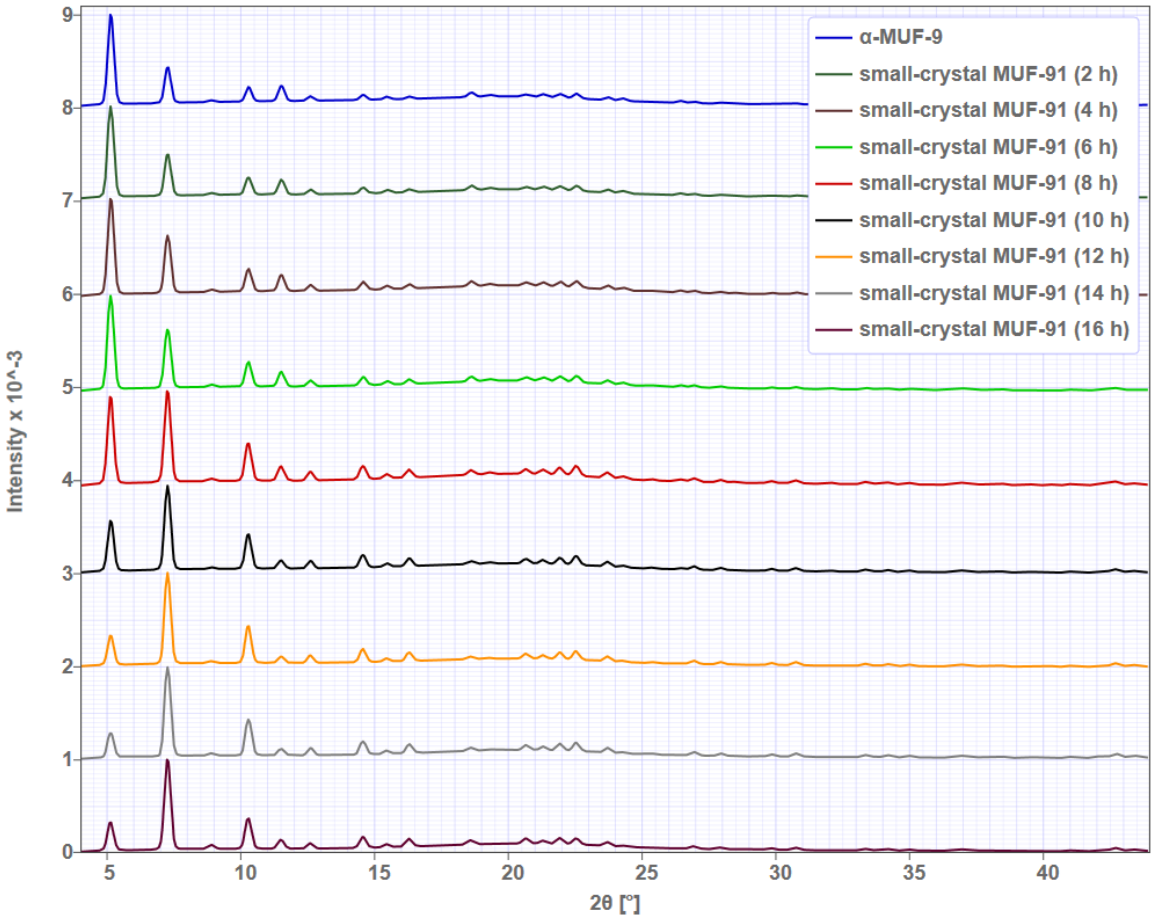


Figure S4. PXRD patterns of microcrystalline MUF-91.

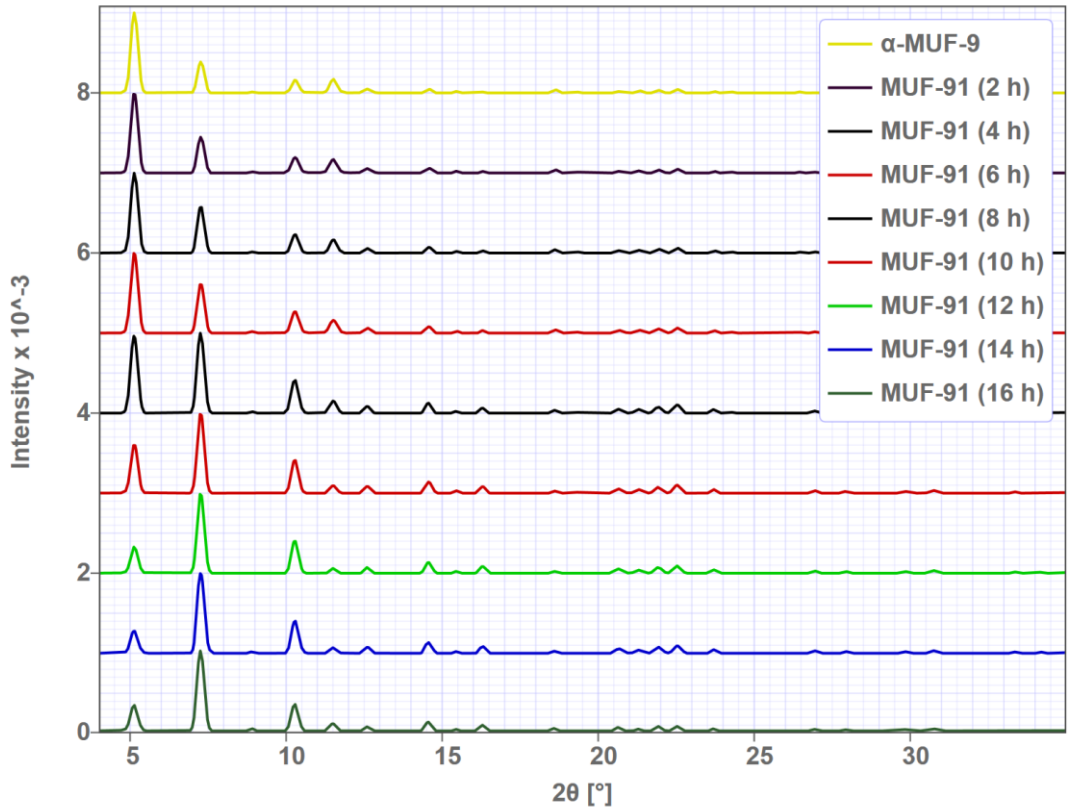


Figure S5: Baseline-corrected and scaled PXRD patterns of microcrystalline MUF-91 as used for interpenetration percentage (PIP%) determination.

S2.2.6.3. *Microcrystalline MUF-91 PIP plot*

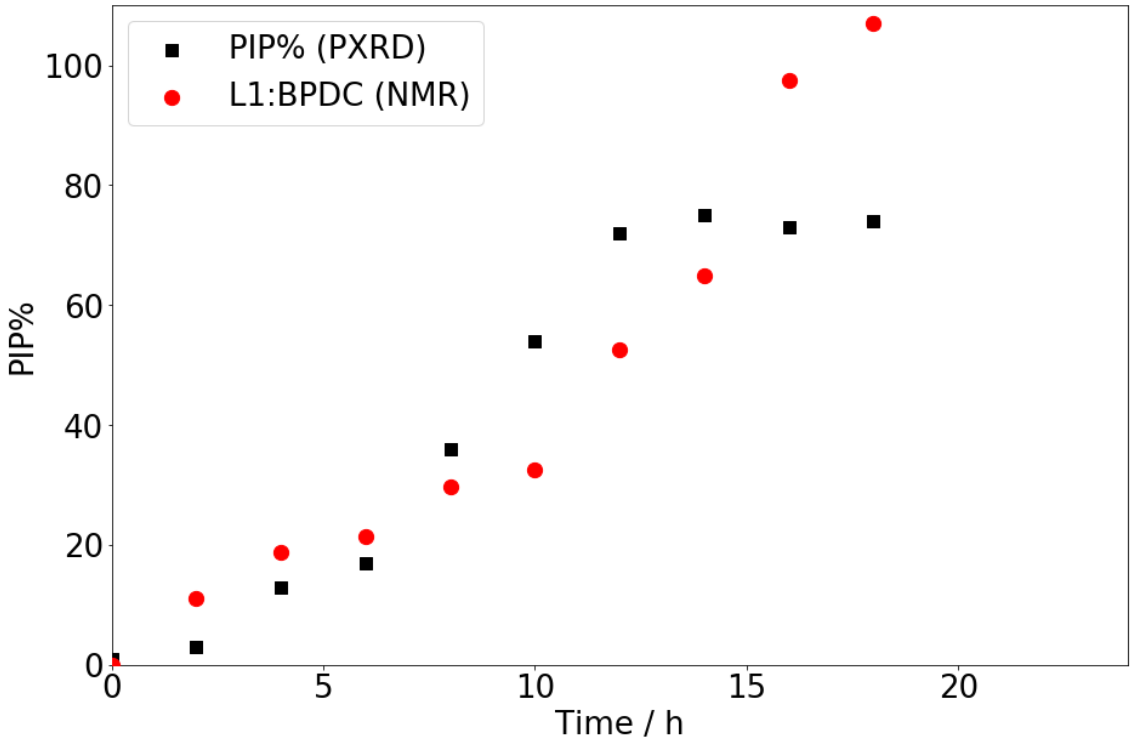
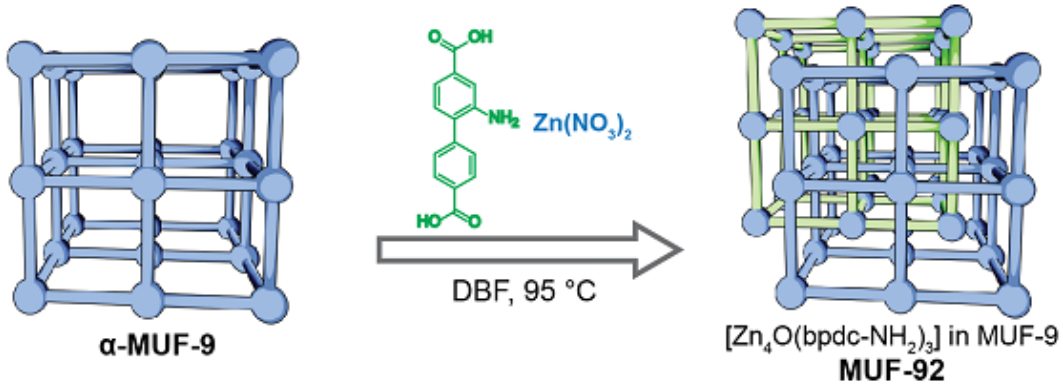


Figure S6: Growth of the interpenetrating sublattice in MUF-91 over time. Black squares represent the interpenetration percentage determined from PXRD patterns and red circles represent the ratio of BPDC to **L1** determined by ^{1}H NMR spectroscopy on digested samples.

S2.3. MUF-92 (MUF-9 interpenetrated by $[Zn_4O(bpdc\text{-}NH_2)_3]$)

S2.3.1. Synthesis of MUF-92 single crystals



α-MUF-9 was synthesised in a 4 mL glass vial with a phenolic cap by a literature method.[1] This method uses 20 µmol of **L1** and yields 5-8 mg of α-MUF-9. A stock solution of $H_2BPDC\text{-}NH_2$ (0.5 mg mL^{-1}), $Zn(NO_3)_2{\cdot}4H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{1}) was prepared in DBF. The solvent was removed from a vial of α-MUF-9 crystals and replaced with this stock solution, after which the vial was heated in a dry bath at 95 °C. The stock solution was removed and replaced with fresh solution every three hours. After various reaction times, the crystals were removed from the dry bath, cooled to room temperature, and washed several times with DBF.

S2.3.2. X-ray crystallography

MUF-92 crystals grown over short reaction times have low PIP% and diffract poorly by nature, as observed for MUF-9 and MUF-10, due to their inhomogeneity. However, at higher PIP%, good data could be obtained (R_1 < 0.11 with no corrections for solvent scattering and resolution greater than 0.75 Å). In the electron density maps, the phenyl rings of the $bpdc\text{-}NH_2$ in the $[Zn_4O(bpdc\text{-}NH_2)_3]$ lattice are exactly perpendicular, unlike those of **L1** in the host MUF-9 lattice.

Table S3: SCXRD data for representative MUF-92 datasets.

	Low PIP%	Medium PIP%	High PIP%
Identification Code	**MUF-92-3h-42pc**	**MUF-92-6h-65pc**	**MUF-92-12h-71pc**
Growth time	3 hours	6 hours	12 hours
Empirical formula	$C_{101.56}H_{59.28}N_{7.25}O_{18.44}Zn_{5.67}$	$C_{111.48}H_{71.54}N_{7.96}O_{21.51}Zn_{6.62}$	$C_{113.92}H_{73.64}N_{8.14}O_{22.26}Zn_{6.85}$
Interpenetration fraction (PIP%)	0.42	0.65	0.71
Formula weight	2046.75	2299.36	2360.33
Temperature / K		100(2)	
Crystal system		cubic	
Space group		P-43m	
a,b,c / Å	17.146(7)	17.140(3)	17.1190(7)
α, β, γ /°		90	
Volume / $Å^3$	5041(6)	5035(3)	5016.9(6)
Z		1	
ρ_{calc} / g cm^{-3}	0.674	0.758	0.781
µ / mm^{-1}	0.698	0.814	0.845
F(000) / e^-	1037	1167	1198
Radiation		Synchrotron (λ = 0.7108 Å)	
2Θ range for data collection / °	5.312 to 32.93	5.314 to 37.634	5.322 to 52.694
Index ranges	-13 ≤ h ≤ 13, -13 ≤ k ≤ 13, -12 ≤ l ≤ 13	-15 ≤ h ≤ 15, -15 ≤ k ≤ 15, -14 ≤ l ≤ 15	-18 ≤ h ≤ 18, -13 ≤ k ≤ 20, -16 ≤ l ≤ 19
Reflections collected	6213	9105	15680
Independent reflections	558 [R_{int} = 0.0781, R_{sigma} = 0.1066]	795 [R_{int} = 0.0950, R_{sigma} = 0.0373]	1926 [R_{int} = 0.0644, R_{sigma} = 0.0341]

Data/restraints/parameters	558/120/66	795/120/66	1926/120/66
Goodness-of-fit on F^2	1.745	1.633	1.212
Final R indexes [I>=2σ (I)]	R_1 = 0.1840, wR_2 = 0.3853	R_1 = 0.1504, wR_2 = 0.3405	R_1 = 0.1098, wR_2 = 0.2979
Final R indexes [all data]	R_1 = 0.1987, wR_2 = 0.4146	R_1 = 0.1580, wR_2 = 0.3560	R_1 = 0.1495, wR_2 = 0.3300
Largest diff. peak/hole / e $Å^{-3}$	0.56/-0.48	0.89/-0.39	0.66/-0.52
Flack parameter	0.55(6)	0.27(4)	0.578(18)

S2.3.3. ^{1}H NMR spectroscopy

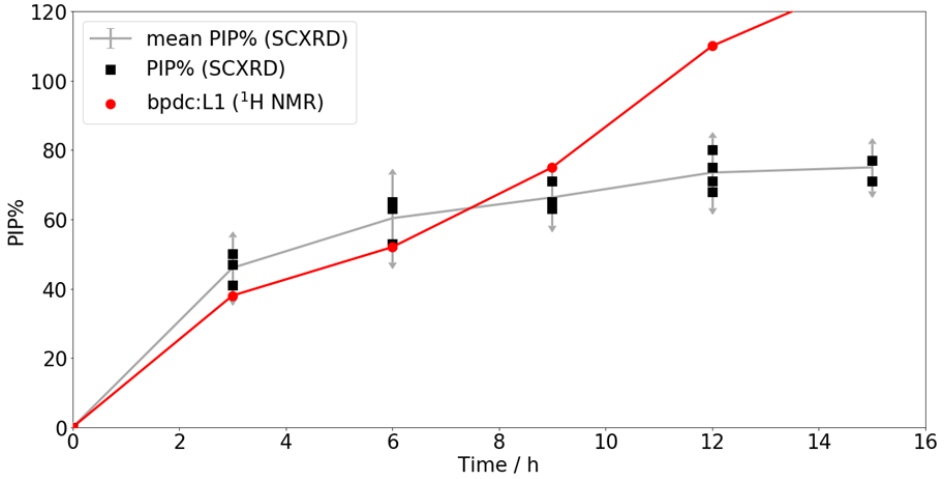


Figure S7: Plot of complementary characterisation data for MUF-92. Black squares represent the PIP% values obtained from SCXRD datasets collected at various time points. Red circles represent the ratio of BPDC-NH_2 to **L1** as determined by ^{1}H NMR spectroscopy. The grey line is drawn through the mean PIP% (from all SCXRD datasets) and the error bars correspond to the 95% confidence interval for the mean PIP% value for the sample.

S2.3.4. PXRD

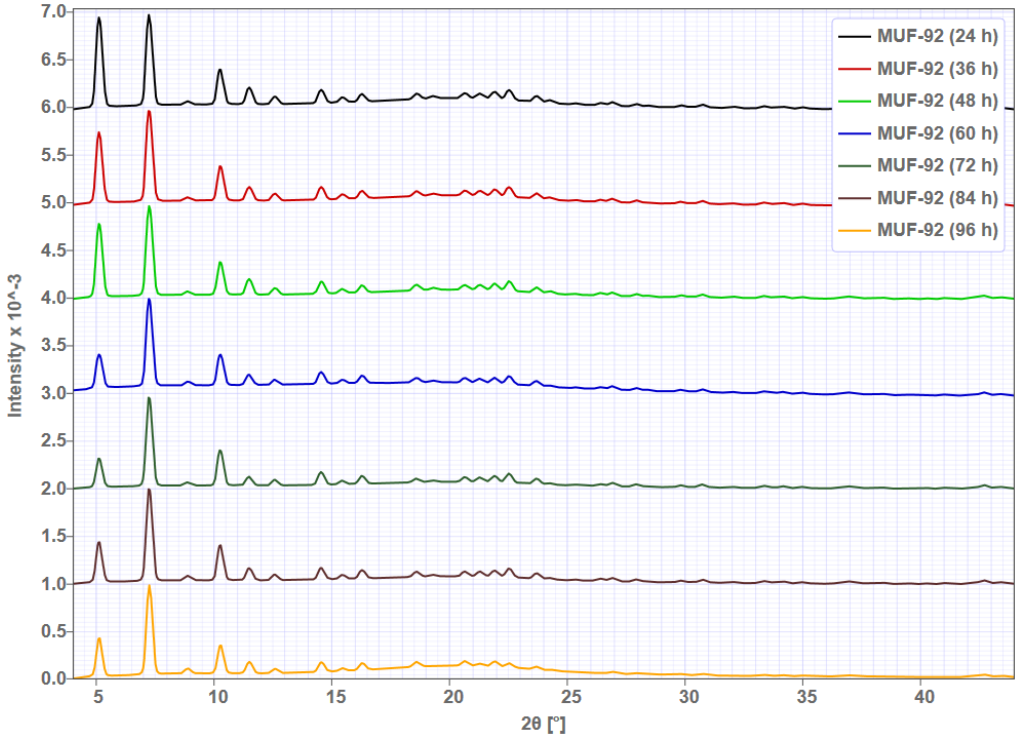


Figure S8: PXRD patterns of MUF-92 after various growth times.

S2.3.5. Optical microscopy

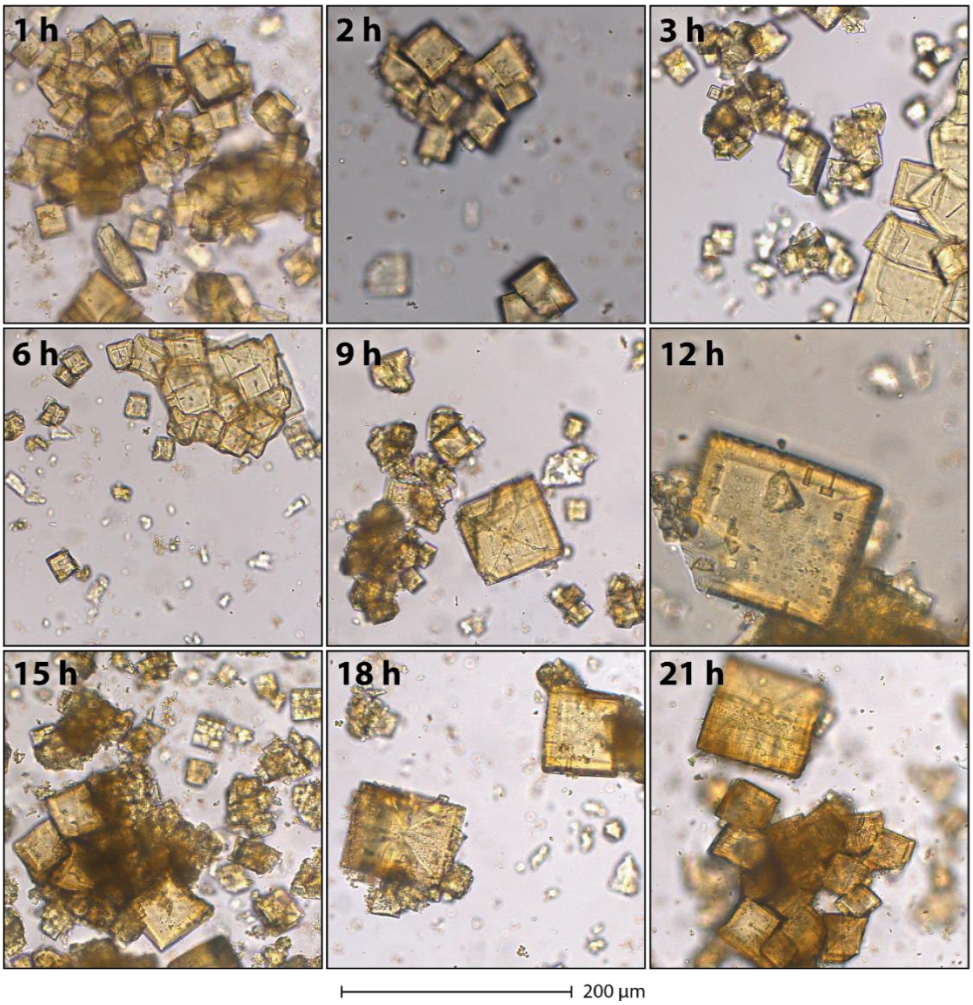


Figure S9. Optical micrographs of MUF-92 at various stages of growth.

S2.3.6. Microcrystalline MUF-92

S2.3.6.1. *Synthesis*

H_2**L1** (36 mg) and $Zn(NO_3)_2 \cdot 4H_2O$ (64 mg) were dissolved in DBF (4 mL) with H_2O (10 µL) in a 25 mL Schott bottle and heated in an 85°C oven for 6 hours, shaking the mixture every hour. The resulting small crystals were then transferred to a 4 mL vial which was centrifuged at 1000 rpm for 0.5 mins. The supernatant was replaced with 2 mL of a stock solution of H_2BPDC-NH_2 (1 mg mL^{-1}), $Zn(NO_3)_2 \cdot 4H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{-1}) in DBF, and the crystals were heated in a dry bath set at 95 °C. At intervals of two hours, the crystals were centrifuged, a sample taken out for analysis, the solution exchanged for fresh stock solution, then heating continued.

S2.3.6.2. *PXRD*

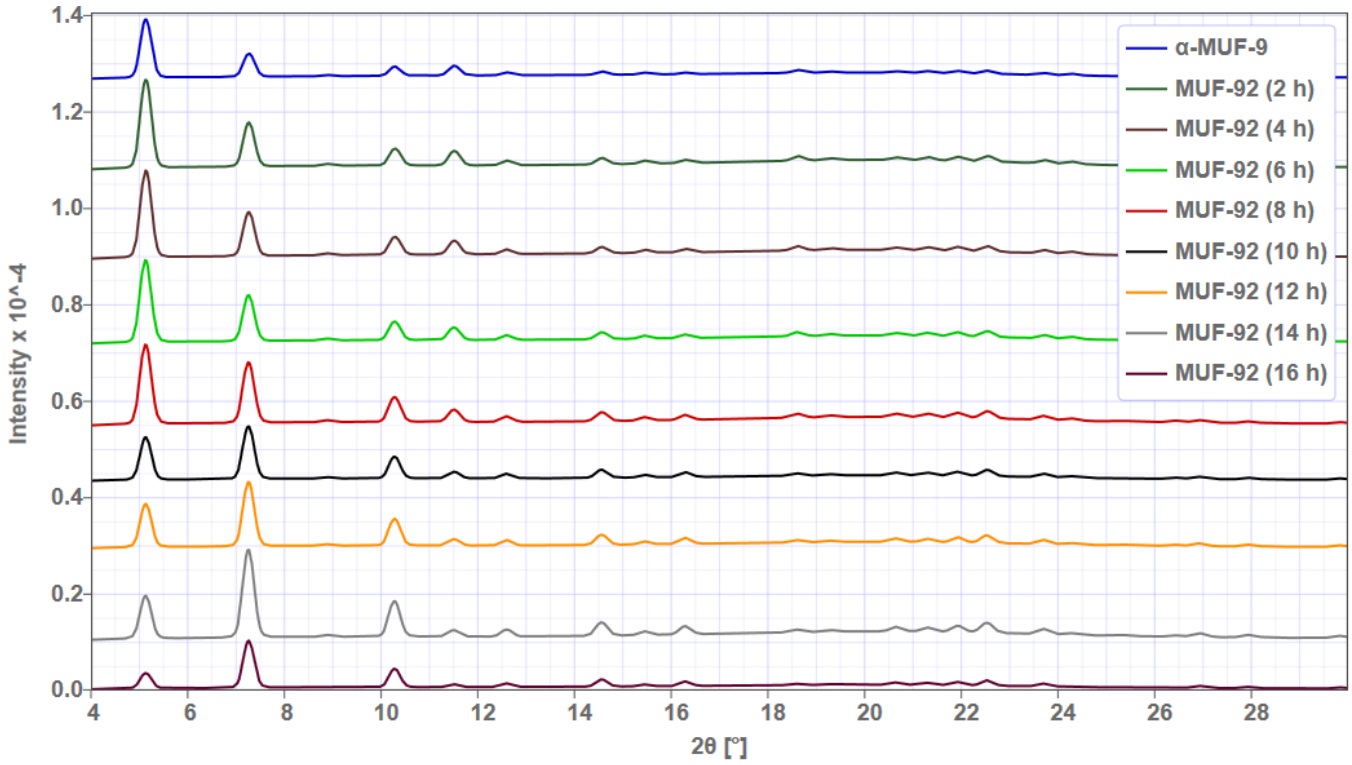


Figure S10: PXRD patterns of microcrystalline MUF-92.

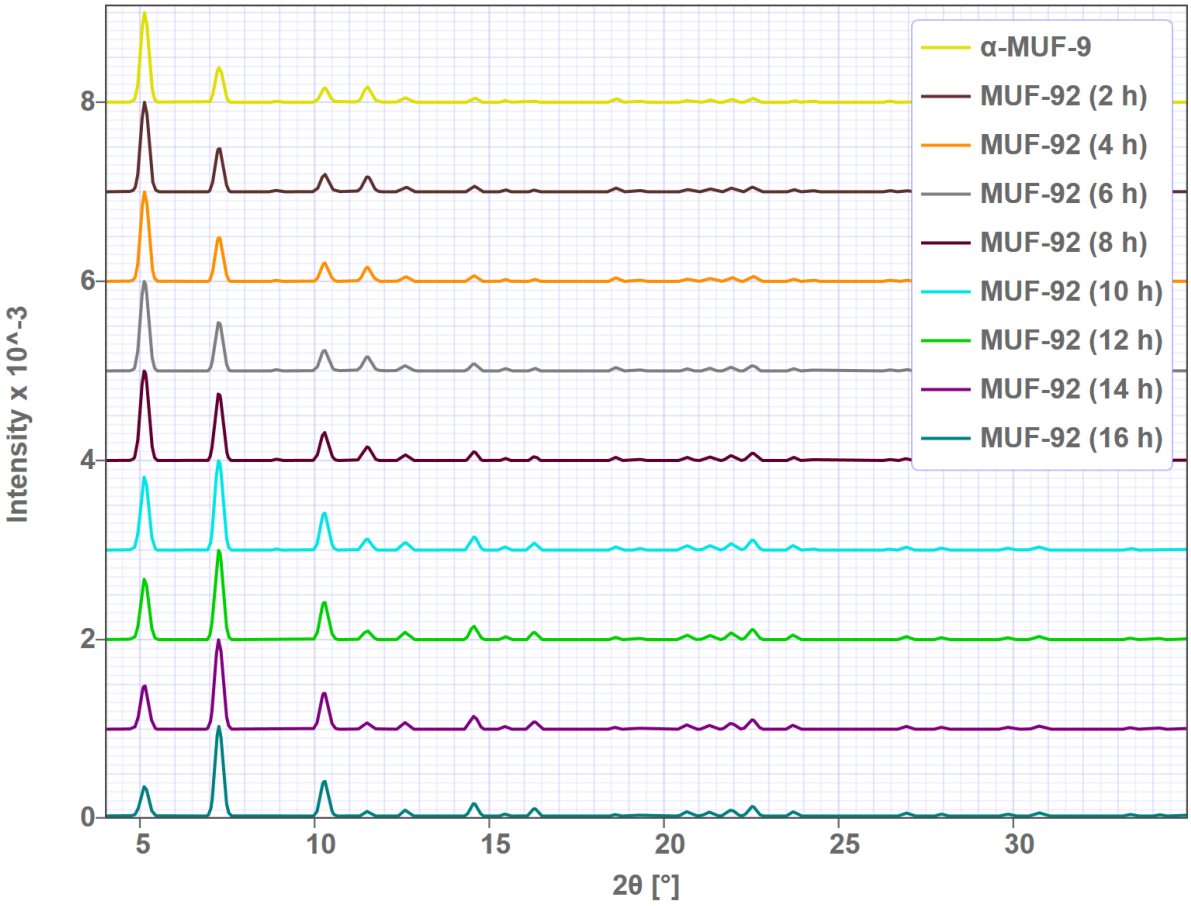


Figure S11: Baseline-corrected and scaled PXRD patterns of microcrystalline MUF-92 as used for PIP% determination.

S2.3.6.3. *Microcrystalline MUF-92 PIP plot*

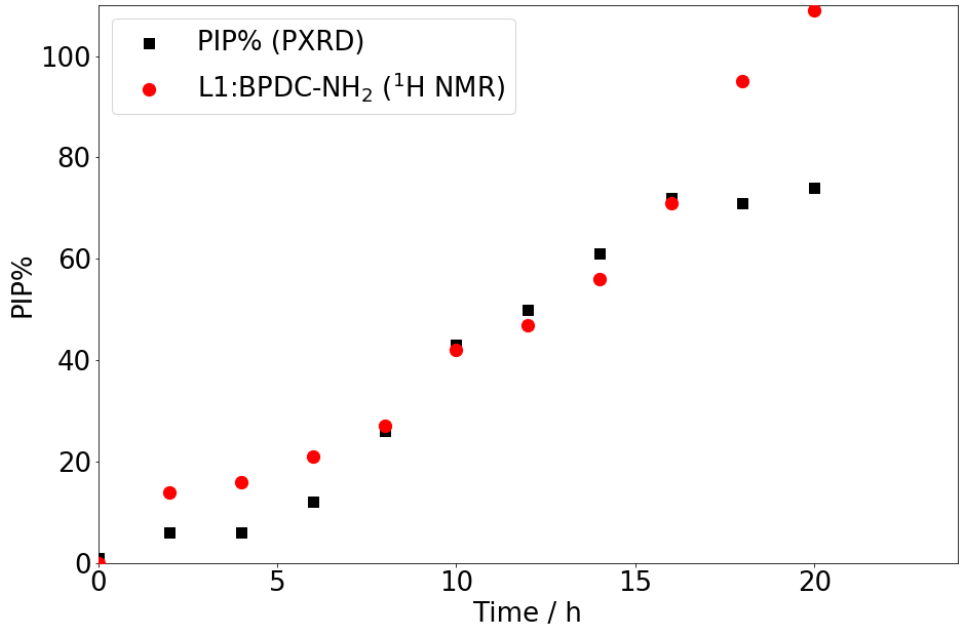
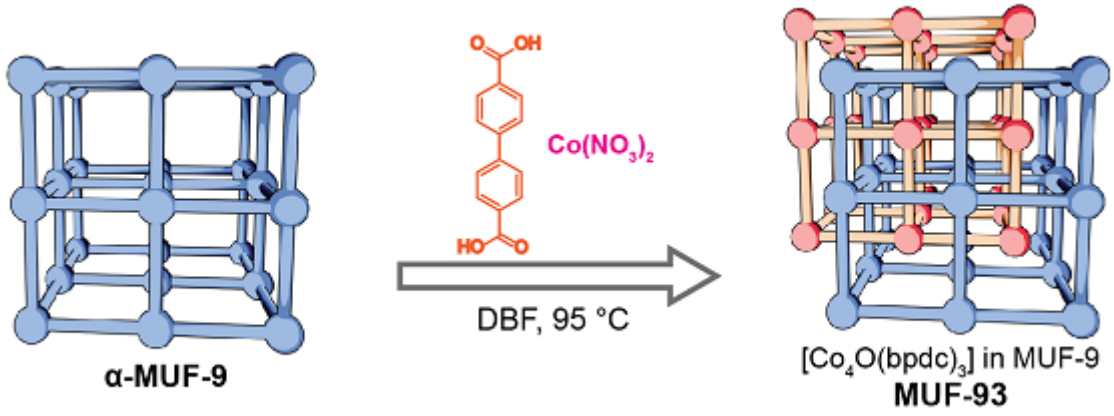


Figure S12: Growth of the interpenetrating sublattice in MUF-92 over time. Black squares represent the PIP% determined from PXRD patterns (Figure S11) and red circles represent the PIP% determined from the BPDC-NH_2 to **L1** ratio in the ^{1}H NMR spectra of digested samples.

S2.4. MUF-93 (MUF-9 interpenetrated by $[Co_4O(bpdc)_3]$)

S2.4.1. Synthesis of MUF-93 single crystals



α-MUF-9 was synthesised in a 4 mL glass vial with phenolic cap by a literature method[1]. A stock solution of H_2-BPDC (0.5 mg mL^{-1}), $Co(NO_3)_2 \cdot 6H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (2 mg mL^{1}) was prepared in DBF. The solvent was removed from a vial of α-MUF-9 and replaced with this stock solution, after which the vial was heated in a dry bath set to 75 °C. The stock solution was removed and replaced with fresh solution every 12 hours. At the desired stage of growth, the vial was removed from the dry bath, cooled to room temperature, and the crystals washed several times with DBF.

S2.4.2. SCXRD

MUF-93 crystals grown over short reaction times have low PIP% and diffract poorly by nature, as observed for MUF-9 and MUF-10, due to their inhomogeneity. However, at higher PIP%, good data could be obtained (R_1 < 0.15 with no corrections for solvent scattering and resolution better than 0.85 Å).

Table S4: SCXRD data for representative MUF-93 datasets.

	Low PIP%	**Medium PIP%**	**High PIP%**
Identification code	MUF-93-12h-4pc	MUF-93-48h-48pc	MUF-93-84h-66pc
Growth time	12 hours	48 hours	84 hours
Empirical formula	$C_{85.71}H_{48.99}Co_{0.16}N_6O_{13.53}Zn_4$	$C_{104.15}H_{59.52}Co_{1.92}N_6O_{19.24}Zn_4$	$C_{111.57}H_{63.76}Co_{2.63}N_6O_{21.53}Zn_4$
Interpenetration fraction	0.04	0.48	0.66
Formula weight	1650.19	2077.31	2249.23
Temperature / K	100(2)	100(2)	100(2)
Crystal system	cubic	cubic	cubic
Space group	P-43m	P-43m	P-43m
a,b,c / Å	16.90(5)	17.101(8)	17.098(8)
α, β, γ /°	90	90	90
Volume / Å3	4827(43)	5001(7)	4998(7)
Z	1.00008	1.00008	1
ρ_{calc} / g cm^{-3}	0.568	0.690	0.747
μ / mm^{-1}	0.532	0.662	0.723
F(000) / e^-	838.0	1052.0	1138.0
Radiation	Synchrotron (λ = 0.71092 Å)	Synchrotron (λ = 0.7085 Å)	
2Θ range for data collection / °	5.392 to 27.312	7.126 to 48.82	5.31 to 46.296
Index ranges	-11 ≤ h ≤ 11, -11 ≤ k ≤ 11, -11 ≤ l ≤ 11	-19 ≤ h ≤ 19, -19 ≤ k ≤ 14, -9 ≤ l ≤ 19	-18 ≤ h ≤ 18, -18 ≤ k ≤ 14, -11 ≤ l ≤ 18
Reflections collected	5274	14970	13070
Independent reflections	319 [R_{int} = 0.1635, R_σ = 0.0493]	1605 [R_{int} = 0.0714, R_σ = 0.0316]	1391 [R_{int} = 0.0513, R_σ = 0.0235]

Data/restraints/parameters	319/223/55	1605/223/55	1391/223/55
Goodness-of-fit on F^2	1.882	2.091	1.715
Final R indexes [I>=2σ (I)]	R1 = 0.2221, wR2 = 0.4955	R1 = 0.1615, wR2 = 0.4415	R1 = 0.1304, wR2 = 0.3709
Final R indexes [all data]	R_1 = 0.2604, wR_2 = 0.5260	R_1 = 0.1807, wR_2 = 0.4646	R_1 = 0.1415, wR_2 = 0.3843
Largest diff. peak/hole / e $Å^{-3}$	0.36/-0.23	2.14/-1.27	1.12/-0.61
Flack parameter*	0.42(7)	0.195(17)	0.307(14)

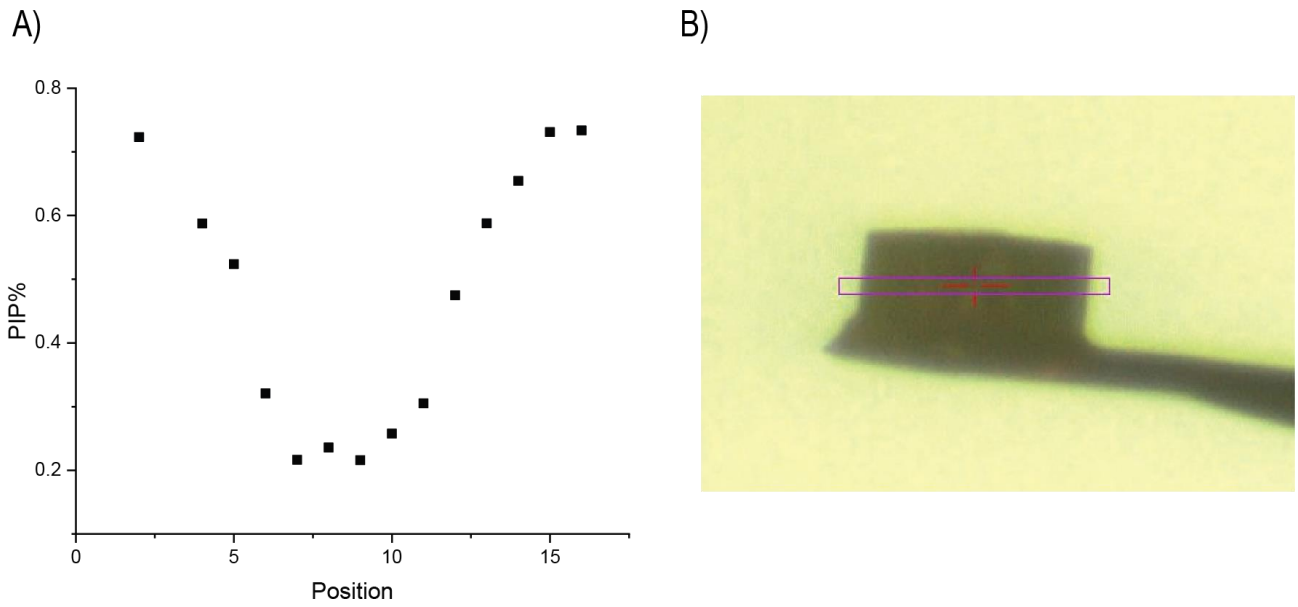


Figure S13: A) A plot of PIP values determined from SCXRD datasets collected at various regions of a single crystal of MUF-93. B) A photograph showing the MUF-93 crystal and the region used to collect the data.

S2.4.3. Atomic Absorption (AAS) and ^{1}H NMR Spectroscopy

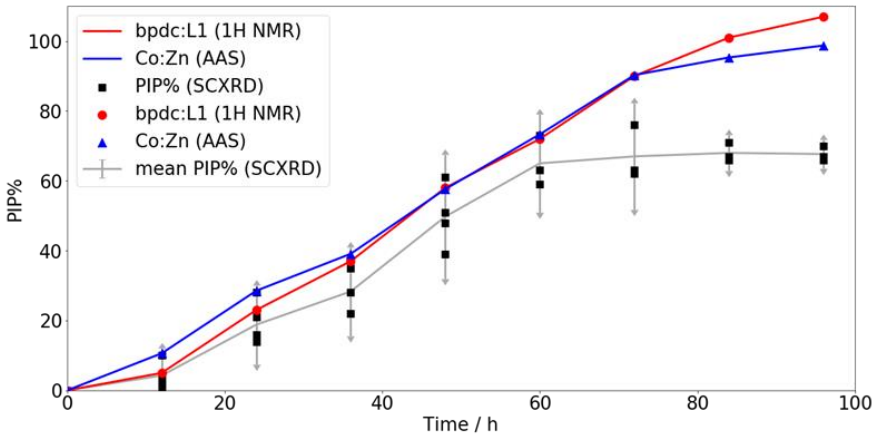


Figure S14: Plot of complementary characterisation data for MUF-93. Black squares represent the PIP% values obtained from SCXRD datasets collected at various time points in the growth of MUF-93. Red circles represent the PIP% determined from the ratio of BPDC to L1 in the ^{1}H NMR spectra of digested samples, and blue triangles represent the Co:Zn ratio, as determined by AAS, in the same digested samples of MUF-93. The grey line is drawn through the mean PIP% (from all SCXRD datasets) and the error bars correspond to the 95% confidence interval for the mean PIP% value for the sample.

S2.4.4. Optical Microscopy

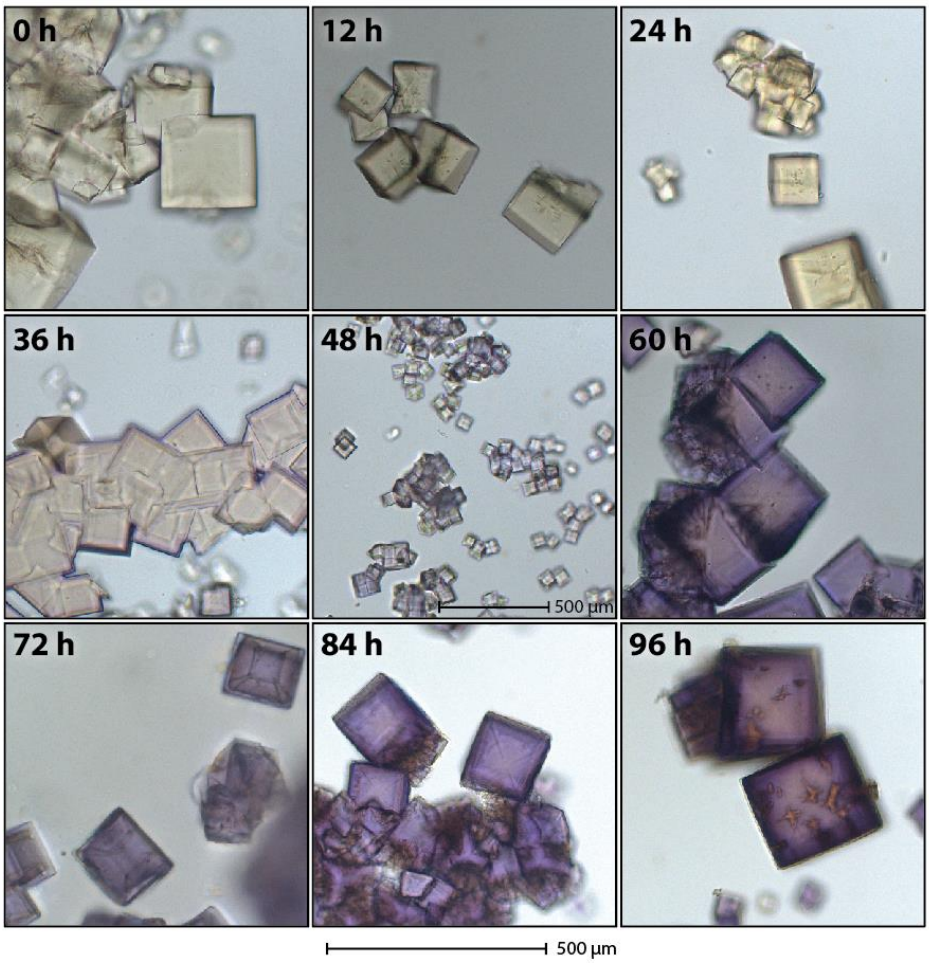


Figure S15. Optical micrographs of MUF-93 after various reaction times.

S2.4.5. Microcrystalline MUF-93

S2.4.5.1. *Synthesis*

H_2**L1** (36 mg) and $Zn(NO_3)_2 \cdot 4H_2O$ (64 mg) were dissolved in DBF (4 mL) with H_2O (10 µL) in a 25 mL Schott bottle and heated in an 85°C oven for 6 hours, shaking the mixture every hour. The resulting small crystals were then transferred to a 4 mL vial which was centrifuged at 1000 rpm for 0.5 mins. The supernatant was replaced with 2 mL of a stock solution of H_2-BPDC-NH_2 (1 mg mL^{-1}), $Co(NO_3)_2 \cdot 6H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{-1}) in DBF, and the crystals were heated in a dry bath set at 95 °C. At intervals of two hours, the crystals were centrifuged, a sample taken out for analysis, the solution exchanged for fresh stock solution, then heating continued.

S2.4.5.2. *Microcrystalline MUF-93 PIP plot*

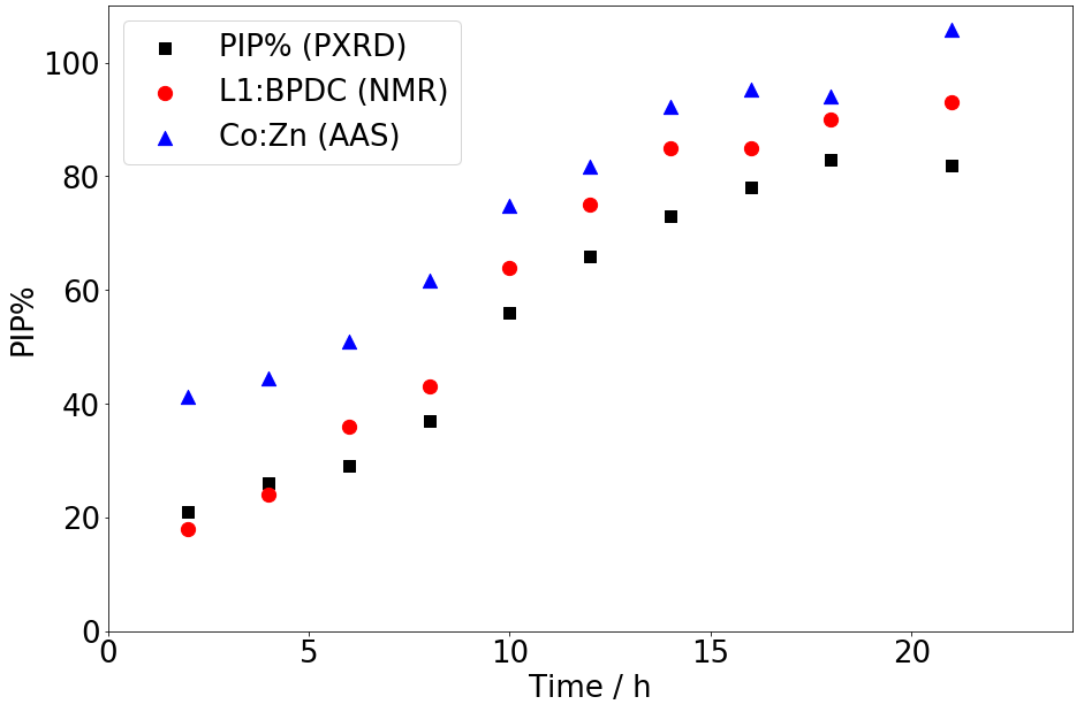


Figure S16: Growth of the interpenetrating sublattice in MUF-93 over time, as determined by PXRD (section 3.1.1), blue triangles represent the Co:Zn ratio, as determined by flame AAS, and red circles represent the ratio of BPDC:L1 as determined by ^{1}H NMR spectroscopy in digested samples.

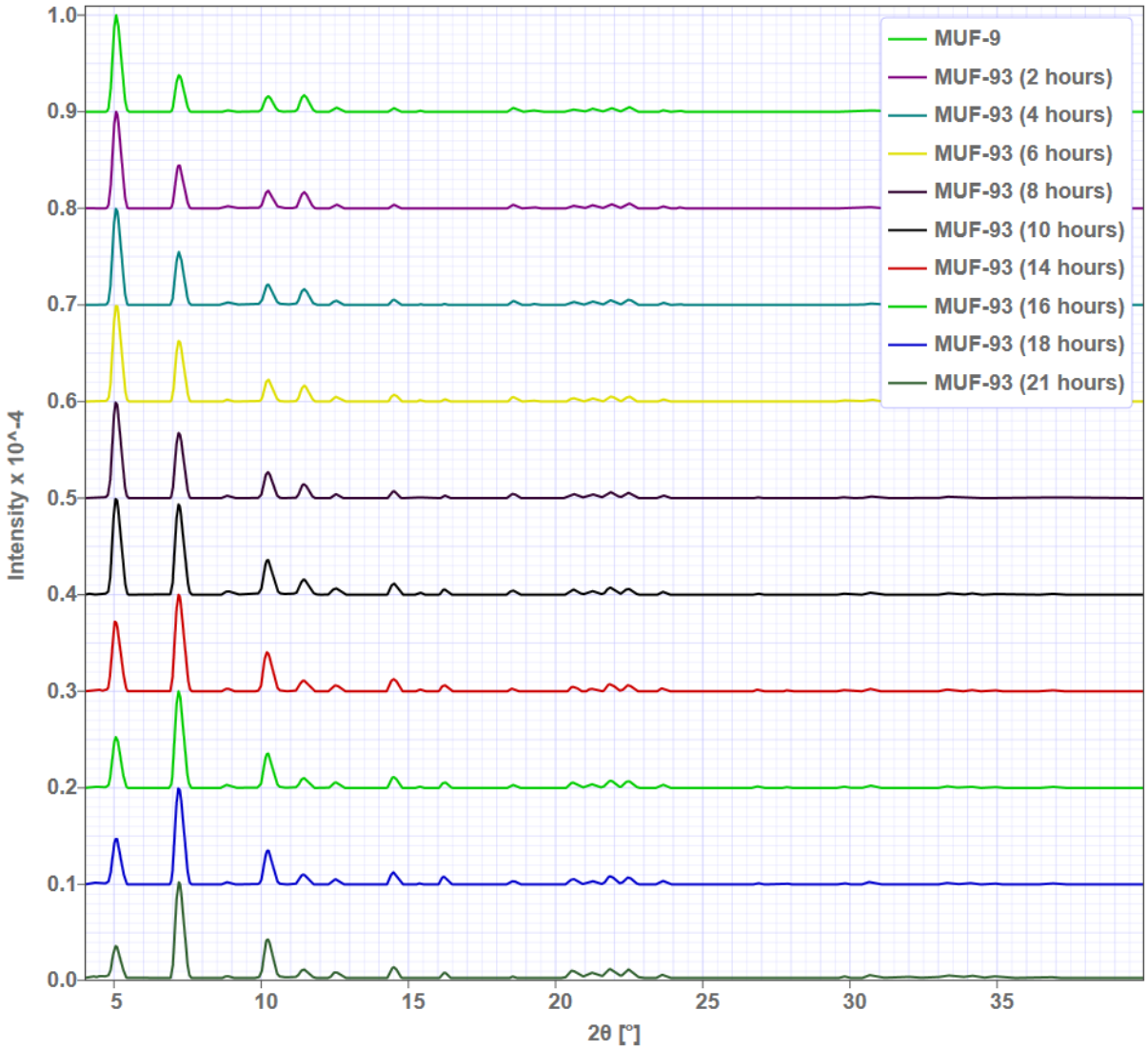


Figure S17: Baseline-corrected and scaled PXRD patterns of microcrystalline MUF-93 as used for interpenetration percentage (PIP%) determination**.**

S3. Site-specific metal identification in MUF-93

S3.1. Processing and analysis of multi-wavelength SCXRD data

S3.1.1. Determination of appropriate wavelength

An X-ray fluorescence excitation scan on a single crystal of MUF-93 was acquired at the Australian Synchrotron MX1 beamline.[4] Wavelengths for SCXRD analysis were chosen at 7500 eV and 9670 eV, close to the absorption edges of cobalt (K β peak) and zinc (where anomalous dispersion contributions become relatively large). A wavelength of 17440 eV – far from an absorption peak of any element in the material – was also used for diffraction.

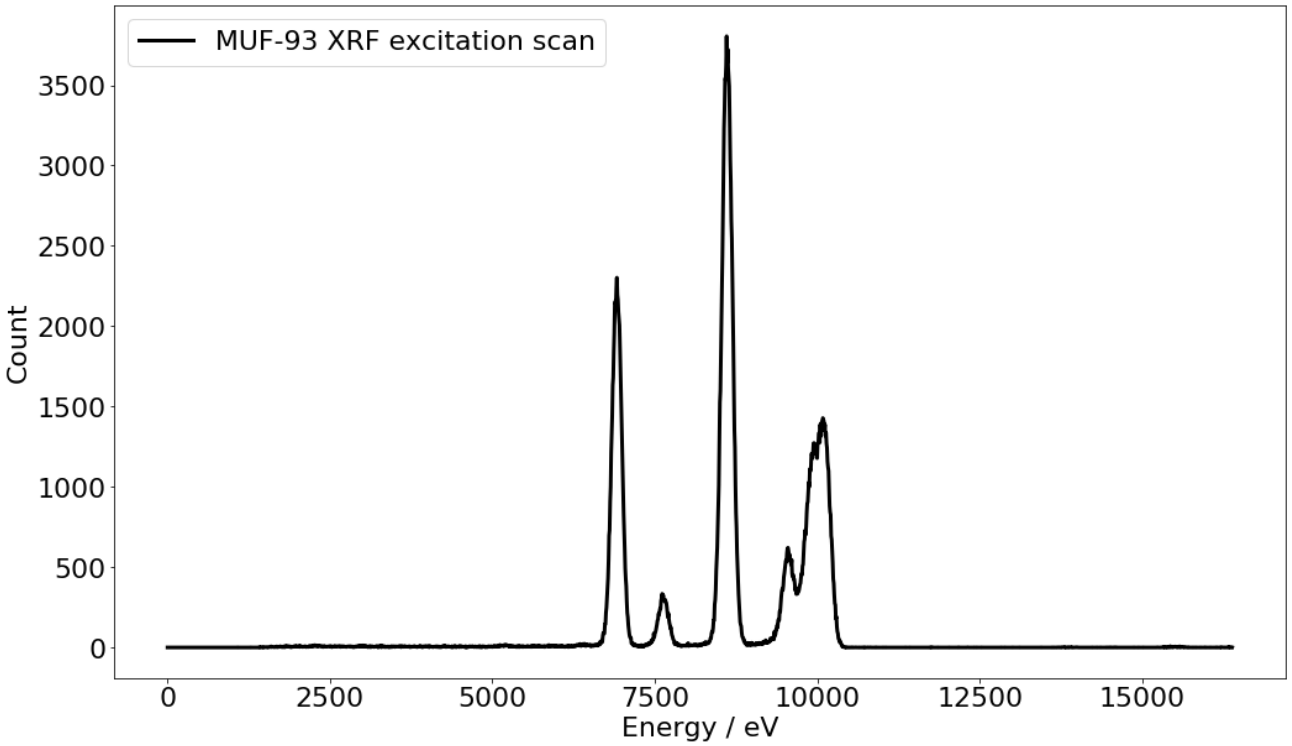


Figure S18: X-ray fluorescence excitation scan of a single crystal of MUF-93.

S3.1.2. Generation of difference datasets and Patterson maps

Datasets were collected on single crystals at wavelengths of 7500 eV or 9670 eV and 17440 eV. The two datasets for each crystal were processed in *XDS*[9] XDS_ASCII.HKL_p1 format, and then using a simple Python script making use of the *CCTBX*[10] toolkit.

The datasets were interpreted with a cubic cell of a = b = c = 17.1 Å and α = β = γ = 90 ° (as determined by the high-resolution datasets) and the *P*-43m space group. The symmetry-equivalent reflections were merged, and the intensities scaled to have the same average. The difference between the two datasets was then calculated. The result was outputted as a SHELX[7]-format HKLF4 file, as well as the derived Patterson map as an CCP4 format map file. The Python code for this is appended in Section S9.3. The Python libraries *Gemmi* (crystallographic data I/O for converting the CCP4 map file to a *numpy* array) and *Plotly* were used to generate colour-mapped 2D slices from the anomalous difference Patterson map, while *pymol* was used for 3D visualisations of the CCP4 maps.

S3.1.3. Interpretation of the anomalous difference datasets

For the cobalt-containing crystals, the difference HKL file was solved using the direct methods routine in SHELXS to locate the cobalt atom of the cobalt cluster. The choice of cobalt in this case is arbitrary; there is no physical atom represented by this data, only the *difference in the signal* from cobalt between the two datasets. Nonetheless, the refinement of a single atom at this position is stable, showing a clear signal that there is cobalt at this location. A small feature in the difference Fourier synthesis at the putative location of the zinc site in MUF-93 can be observed, but it is impossible to refine an atom at that position with any occupancy. When refined with a solvent mask, which is essentially just background reduction in this case and has no physical meaning, the refinement statistics are good. Since most traditional refinement parameters are not meaningful for these datasets so they are omitted from the table below, but the datasets themselves are available for review as Supporting Information.

In the table below, “Site 1” refers to cluster at the cell corner. “Site 2” refers to the cluster at the cell centre. Electron counts were integrated over a radius of 1.7 Å, the average of the atomic radii of zinc and cobalt. Uncertainties for the exchange percentage were calculated using the RMSD of the electron density map.

Table S5: SHELXL refinement results and calculated statistics for anomalous difference datasets.

Sample	Integrated value at Site 1	Integrated value at Site 2	Ratio	Exchange of Co(II) into Zn(II) sites	R_1	wR_2
MUF-93, at the plateau of secondary growth after 60 hours. PIP = 50%. Difference data between 17440 eV and cobalt edge at 7500 eV	356.0	10.2	1:0.029	2.9 ± 4.7%	3.45 %	14.9 %
MUF-93, beyond the plateau of secondary growth after 168 hours. PIP = 47%. Difference data between 17440 eV and cobalt edge at 7500 eV	540.4	184.0	1:0.34	34 ± 2.6%	1.36 %	5.88 %
Controls						
α-MUF-9 heated in a solution of $Co(NO_3)_2$. Difference data between 17440 eV and cobalt edge at 7500 eV	No peak	Intense peak	N/A	N/A	4.73 %	13.0 %
β-MUF-9, difference data between 17440 eV and zinc edge at 9670 eV	233.7	223.4	1:0.96	N/A	1.97 %	6.45 %

Note: Integrated values at each metal atom site are proportional to the electron count of the specified metal at each site but are not absolute values since F(000) values were not determined for the difference datasets.

Using the *WinGX FFT* utility, cube format electron density maps of F_{obs} using the supplied phases from this model were produced, and slices rendered with *Plotly*:

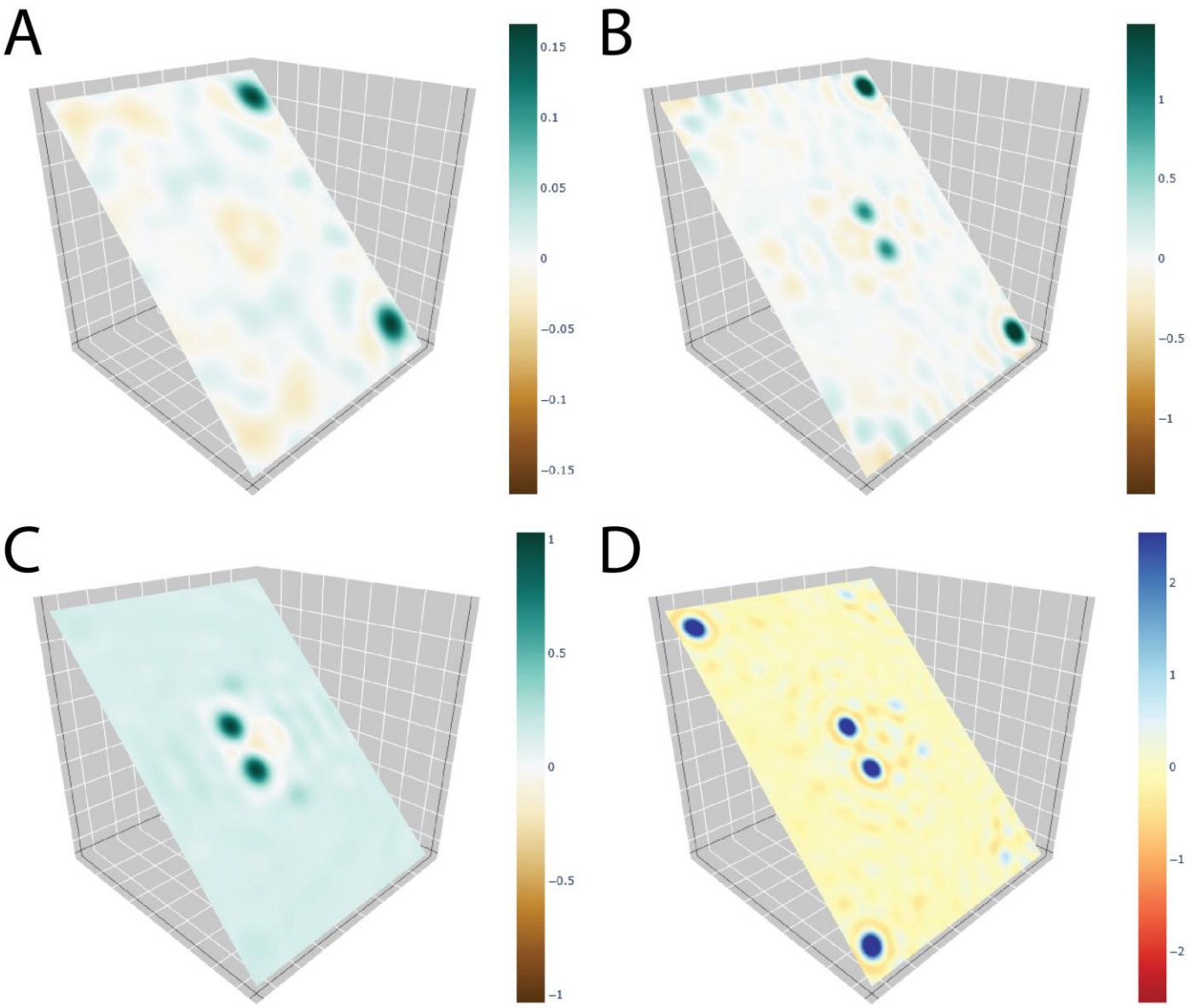


Figure S19: Slices of (Fobs, Φcalc) electron density maps from various difference datasets. (A) MUF-93 at the plateau of secondary growth after 60 hours, difference data between 17440 eV and the cobalt edge at 7500 eV. (B) MUF-93 beyond the plateau of secondary growth after 168 hours, difference data between 17440 eV and the cobalt edge at 7500 eV. (C) α-MUF-9 heated in a 20 mg mL^{-1} cobalt nitrate solution for 108 hours (section S5.2), difference data between 17440 eV and the cobalt edge at 7500 eV. (D) β-MUF-9, difference data between 17440 eV and the zinc edge at 9670 eV.

Table S6: Summary of the files relevant to the anomalous dispersion experiments uploaded as supplementary information.

File name	MOF	Energy / keV	Notes
MUF-93-50pc-60h-Fobs.cube	MUF-93 at plateau of growth	N/A	F_{obs} electron density map of difference data in gaussian cube format.
MUF-93-50pc-60h-7500eV.hkl	MUF-93 at plateau of growth	7.5	Reflection data in SHELX format.
MUF-93-50pc-60h-17500eV.hkl	MUF-93 at plateau of growth	17.5	Reflection data in SHELX format.
MUF-93-50pc-60h-difference.hkl	MUF-93 at plateau of growth	N/A	Reflection data in SHELX format.
MUF-93-50pc-60h-difference.res	MUF-93 at plateau of growth	N/A	Refinement output from SHELX for the cobalt substructure.
MUF-93-46pc-168h-Fobs.cube	MUF-93 beyond plateau of growth	N/A	F_{obs} electron density map of difference data in gaussian cube format.
MUF-93-46pc-168h-7500eV.hkl	MUF-93 beyond plateau of growth	7.5	Reflection data in SHELX format.
MUF-93-46pc-168h-17440eV.hkl	MUF-93 beyond plateau of growth	17.44	Reflection data in SHELX format.
MUF-93-46pc-168h-difference.hkl	MUF-93 beyond plateau of growth	N/A	Reflection data in SHELX format.
MUF-93-46pc-168h-difference.res	MUF-93 beyond plateau of growth	N/A	Refinement output from SHELX for the cobalt substructure.
a-MUF-9-Co-exchanged-Fobs.cube	α-MUF-9, exposed to high concentration cobalt solution	N/A	F_{obs} electron density map of difference data in gaussian cube format.
a-MUF-9-Co-exchanged-7500eV.hkl	α-MUF-9, exposed to high concentration cobalt solution	7.5	Reflection data in SHELX format.
a-MUF-9-Co-exchanged-17500eV.hkl	α-MUF-9, exposed to high concentration cobalt solution	17.5	Reflection data in SHELX format.
a-MUF-9-Co-exchanged-difference.hkl	α-MUF-9, exposed to high concentration cobalt solution	N/A	Reflection data in SHELX format.
a-MUF-9-Co-exchanged-difference.res	α-MUF-9, exposed to high concentration cobalt solution	N/A	Refinement output from SHELX for the cobalt substructure.
B-MUF-9-difference-zinc-Fobs.hkl	β-MUF-9	N/A	F_{obs} electron density map of difference data in gaussian cube format.
B-MUF-9670eV.hkl	β-MUF-9	9.67	Reflection data in SHELX format.
B-MUF-9-17440.hkl	β-MUF-9	17.44	Reflection data in SHELX format.
B-MUF-9-difference-Zn.hkl	β-MUF-9	N/A	Reflection data in SHELX format.
B-MUF-9-difference-Zn.res	β-MUF-9	N/A	Refinement output from SHELX for the zinc substructure.

S4. Synthesis, characterisation and catalytic activity of MUF-101

S4.1. Synthesis of (R)-MUF-101 and (S)-MUF-101

α-MUF-10 was synthesized in a 4 mL glass vial with phenolic cap by a literature method.[1] This used 20 µmol of enantiopure (R)- or (S)-**L1** and yields 5-8 mg of α-(R)-MUF-10 and α-(S)-MUF-10, respectively.

A stock solution of H_2**L2** (1 mg mL^{-1}), $Zn(NO_3)_2 \cdot 4H_2O$ (2 mg mL^{-1}) and 2-fluorobenzoic acid (3 mg mL^{-1}) was prepared in N,N-di-n-propylformamide. α-(R)-MUF-10 or α-(S)-MUF-10 crystals were placed in this stock solution then heated in a dry bath set to 105 °C. The stock solution was removed and replaced with fresh solution every two hours. After eight hours, the crystals were removed from the dry bath, cooled to room temperature, and washed several times with DBF. This yields (R)-MUF-101 (from α-(R)-MUF-10) and (S)-MUF-101 (from α-(S)-MUF-10).

S4.2. ^{1}H NMR spectroscopy and PXRD of MUF-101

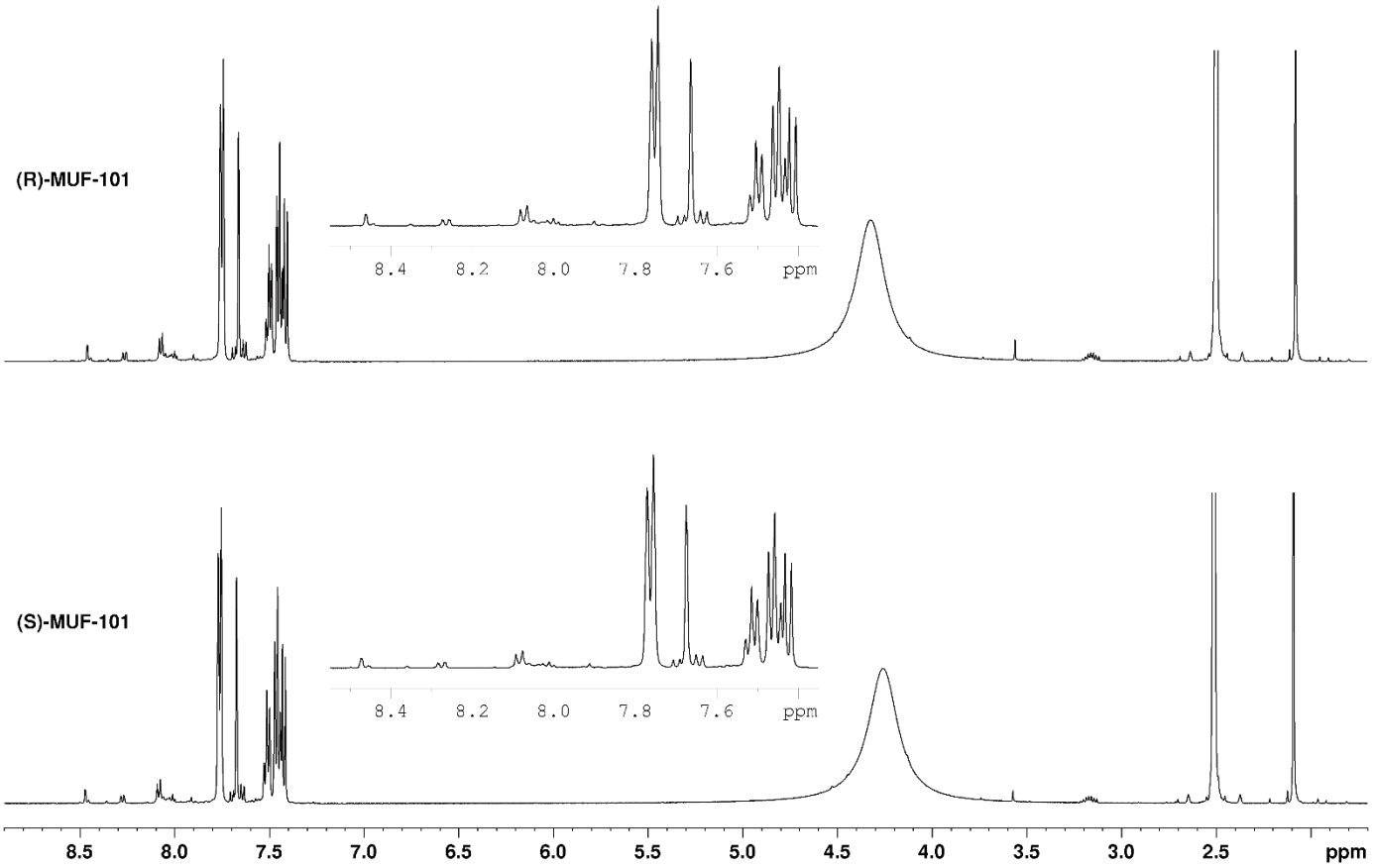


Figure S20: ^{1}H NMR spectra of digested **(R)-**MUF-101 **and (S)-**MUF-101 **in** d_6-DMSO**/**DCl.

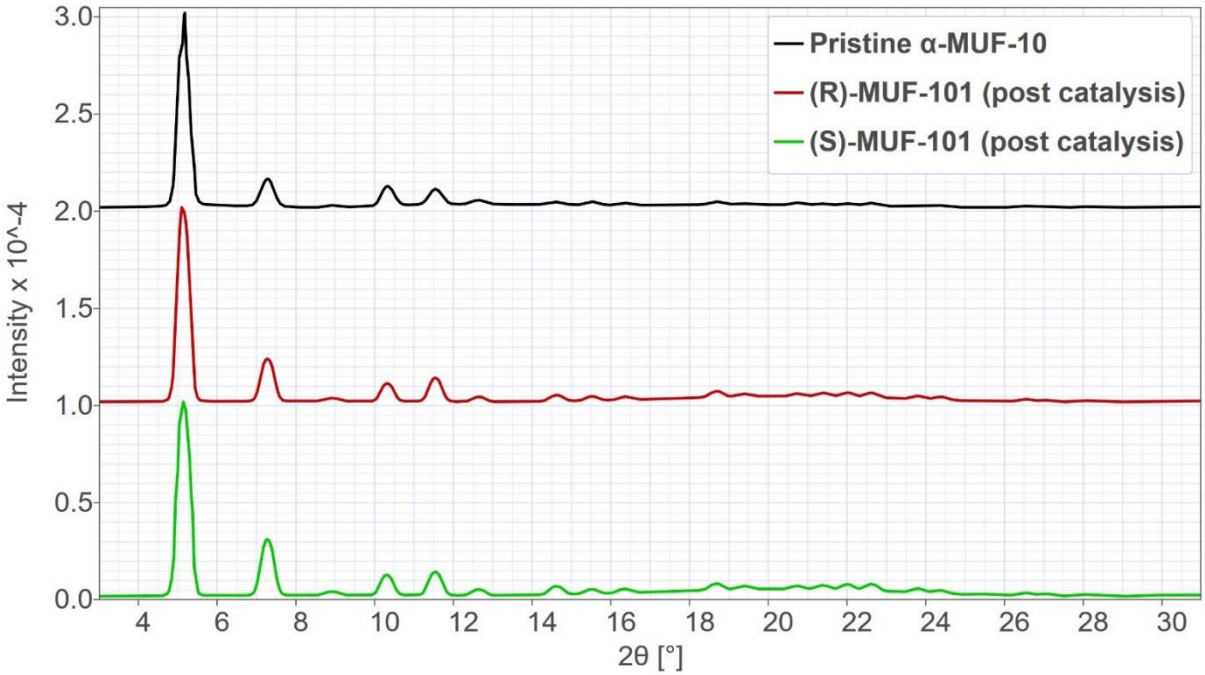
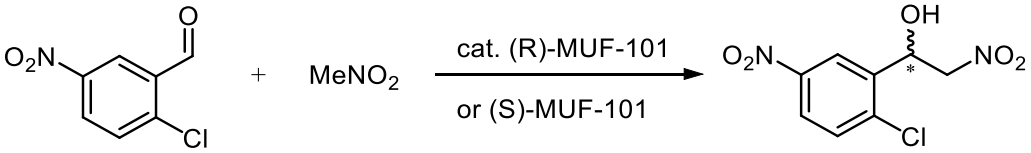


Figure S21: PXRD patterns of MUF-101 following catalysis.

Table S7: The L1:L2 ratio in MUF-101 as deduced by ^{1}H NMR spectroscopy of digested samples.

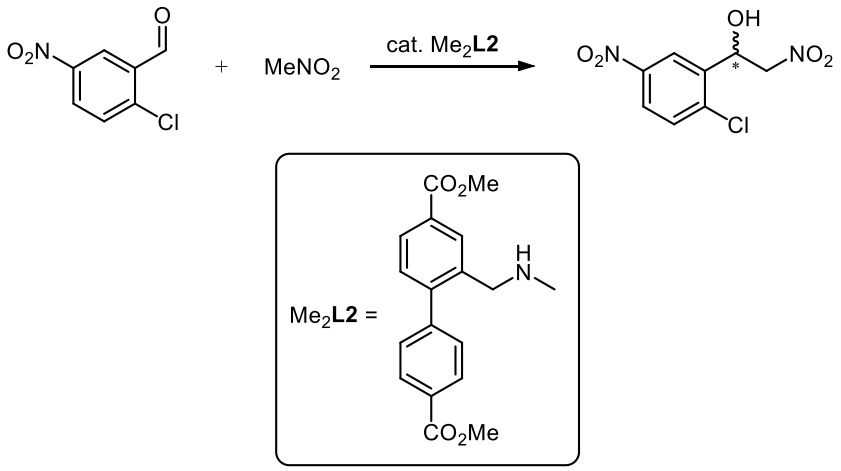
MOF	L1:L2 ratio
(R)-MUF-101	1:0.15
(S)-MUF-101	1:0.15

S4.3. Henry reaction between nitromethane and 2-chloro-5-nitrobenzaldehyde



Scheme S3: Henry reaction between nitromethane and 2-chloro-5-nitrobenzaldehyde catalysed by MUF-101.

Approximately 3 mg of (R)-MUF-101 or (S)-MUF-101 crystals were thoroughly exchanged with DMF and then dioxane. A stock solution (0.6 mL) comprising 2-chloro-5-nitrobenzaldehyde (39 mg, 210 µmol) and $MeNO_3$ (570 µL, 10.5 mmol, 50 eq.) in dioxane (5.4 mL) was then introduced. The reaction mixture was kept at 30 °C for five days and the products were analyzed using HPLC.



Scheme S4: The Henry reaction between nitromethane and 2-chloro-5-nitrobenzaldehyde catalysed by Me$_2$**L2**.

Me$_2$**L2** was used as a homogeneous catalyst to confirm that it was competent towards this reaction. A solution of 2-chloro-5-nitrobenzaldehyde (6.5 mg, 35 µmol) and $MeNO_3$ (94 µL, 1.75 mmol, 50 eq.) were mixed in dioxane (0.9 mL) and Me$_2$**L2** (1.0 mg, 3.2 µmol, 9 % loading) was added. The homogenous mixture was warmed in a dry bath set to 30°C for five days.

S4.3.1. Catalysis HPLC

Reactions were analysed by HPLC with UV detection at 254 nm using a Phenomenex Lux-amylose-1 column and MeCN:H_2O (55/45) as the mobile phase with a flow rate of 0.5 mL min^{-1}. The products were identified by in-line mass spectrometry.

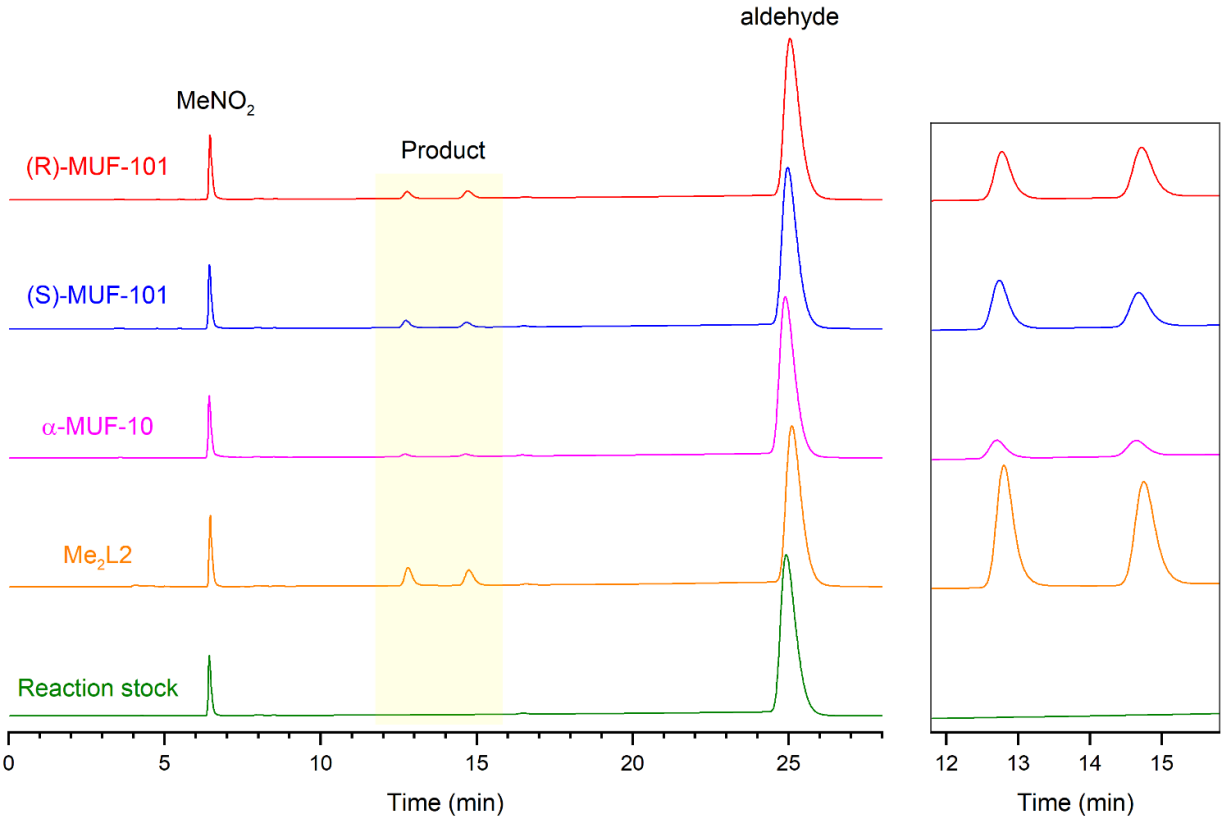


Figure S22: HPLC chromatograms of the Henry reaction between nitromethane and 2-chloro-5-nitrobenzaldehyde using various catalysts.

Table S8: Henry reaction of nitromethane and 2-chloro-5-nitrobenzaldehyde using various catalysts.

Catalyst	**Occupancy of interpenetrating lattice (PIP%)**	**Catalyst loading [a]**	**e.e.[b]**
(R)-MUF-101	15	0.6 %	-9.4
(S)-MUF-101	15	0.6 %	10.2
Control reactions			
α-MUF-10[c]	0	0	0
Me_2**L2**	N/A	9 %	0
No catalyst	N/A	N/A	-

[a] Catalyst loading is the molar ratio of L2 in MUF-101 to 2-chloro-5-nitrobenzaldehyde.
[b] The ee was calculated as the peak area of the peak eluting at 12.7 mins minus the peak area of the peak eluting at 14.9 mins.
[c] A low level of catalytic activity arises from the presence of trace residual DBF.

S5. Control experiments

S5.1. Heating MUF-9 and MUF-10 alone in various solvents

Samples of α-MUF-9 and α-MUF-10 were prepared by a reported procedure[1]. Each sample was washed with a different solvent, then heated in that solvent in a dry bath set to 95 °C to check whether autocatenation occurred.

α-MUF-9 autocatenates in DMF, DEF and DPF but not in DBF. α-MUF-10 autocatenates in DMF but not in DBF nor DPF.

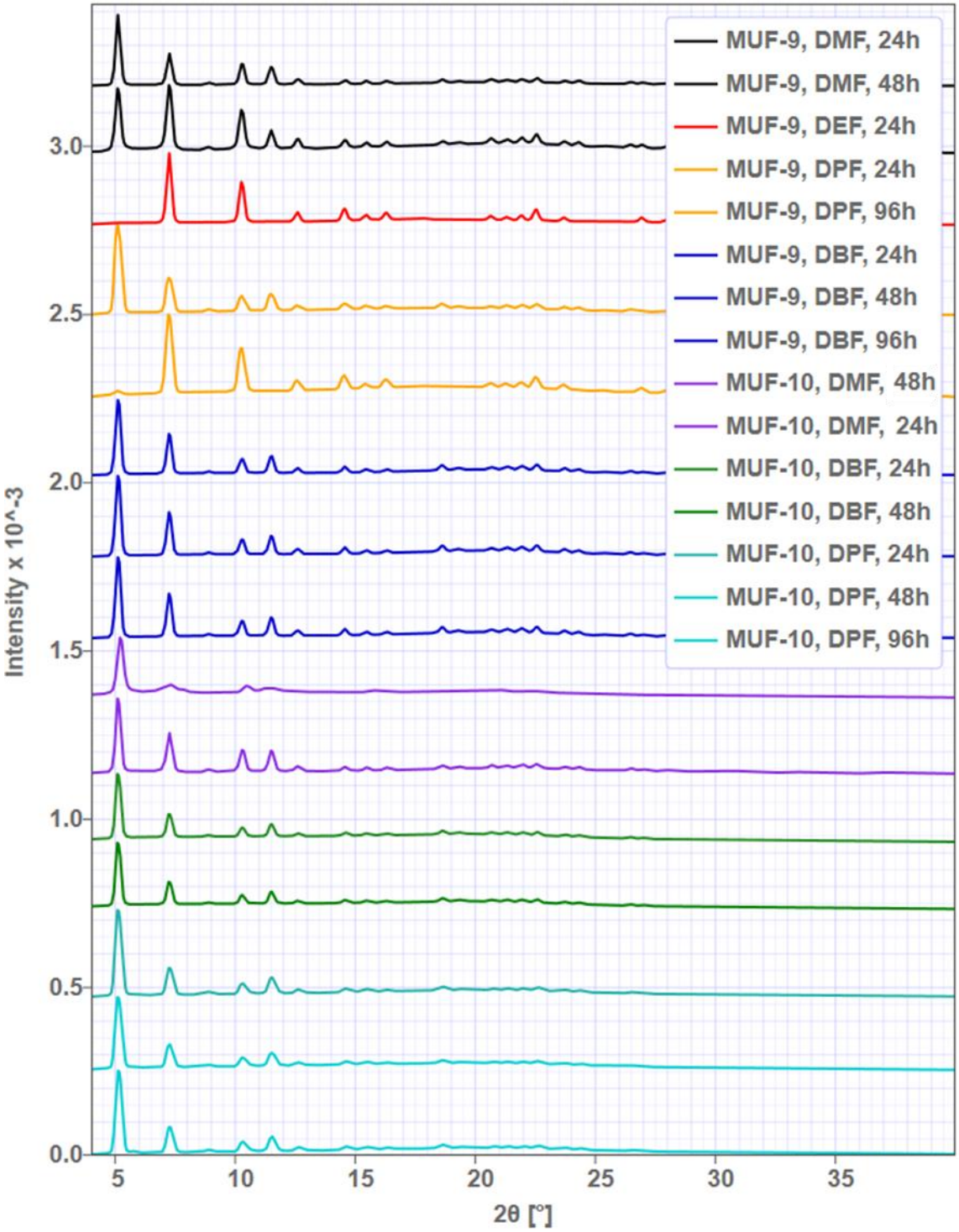


Figure S23: PXRD diffractograms of α-MUF-9 and α-MUF-10 heated in various solvents.

S5.2. Heating MUF-9 with $Co(NO_3)_2$

Samples of α-MUF-9 were exposed to solutions of $Co(NO_3)_2 \cdot 6H_2O$ in DBF at concentrations of 2 mg mL^{-1} and 20 mg mL^{-1}, and heated in each solution in a dry bath set to 95 °C. The samples were analysed by PXRD, then digested with DCl in DMSO and analysed with flame AAS.

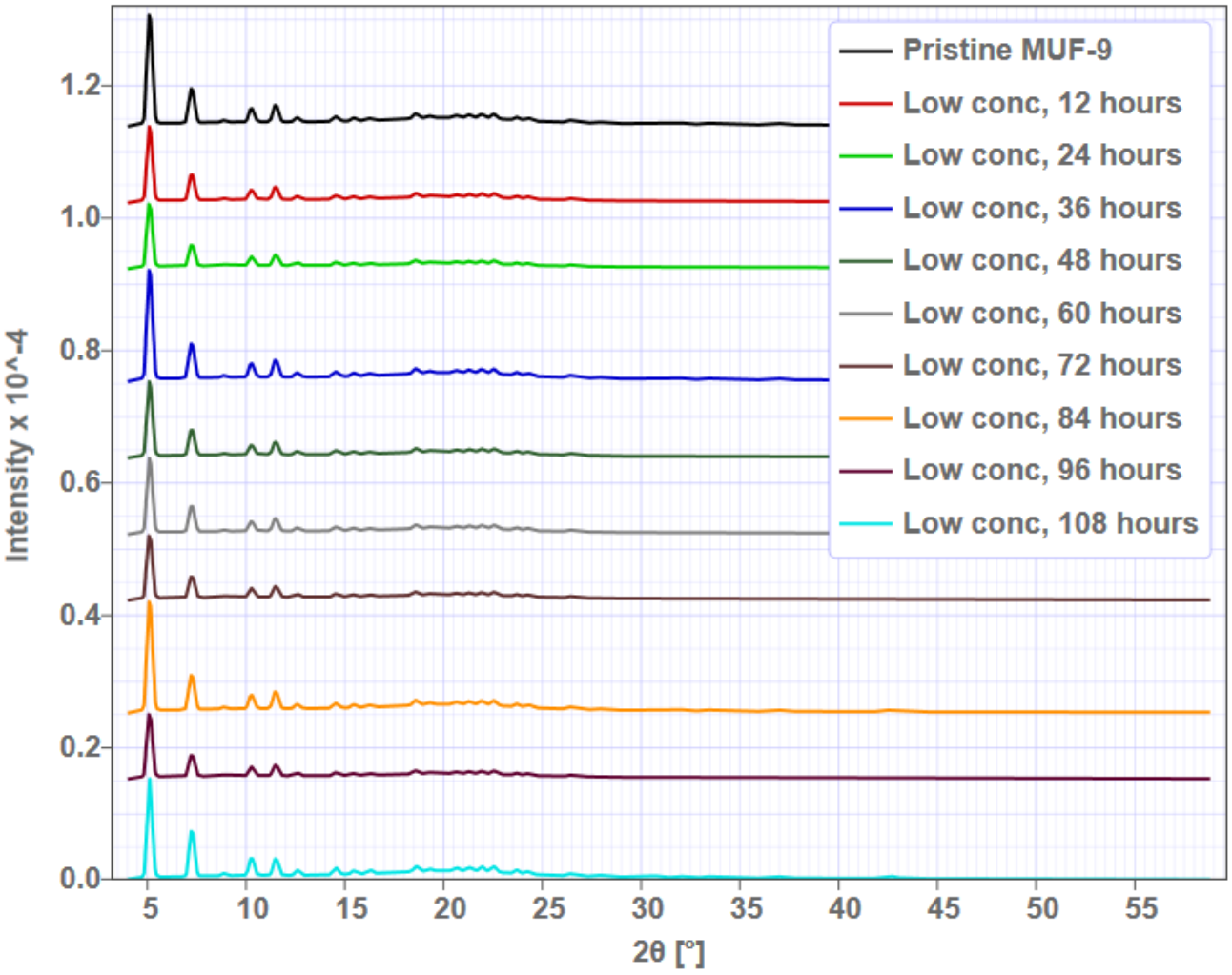


Figure S24: PXRD patterns of α-MUF-9 heated with a 2 mg mL^{-1} solution of $Co(NO_3)_2 \cdot 6H_2O$ in DBF at 95 °C.

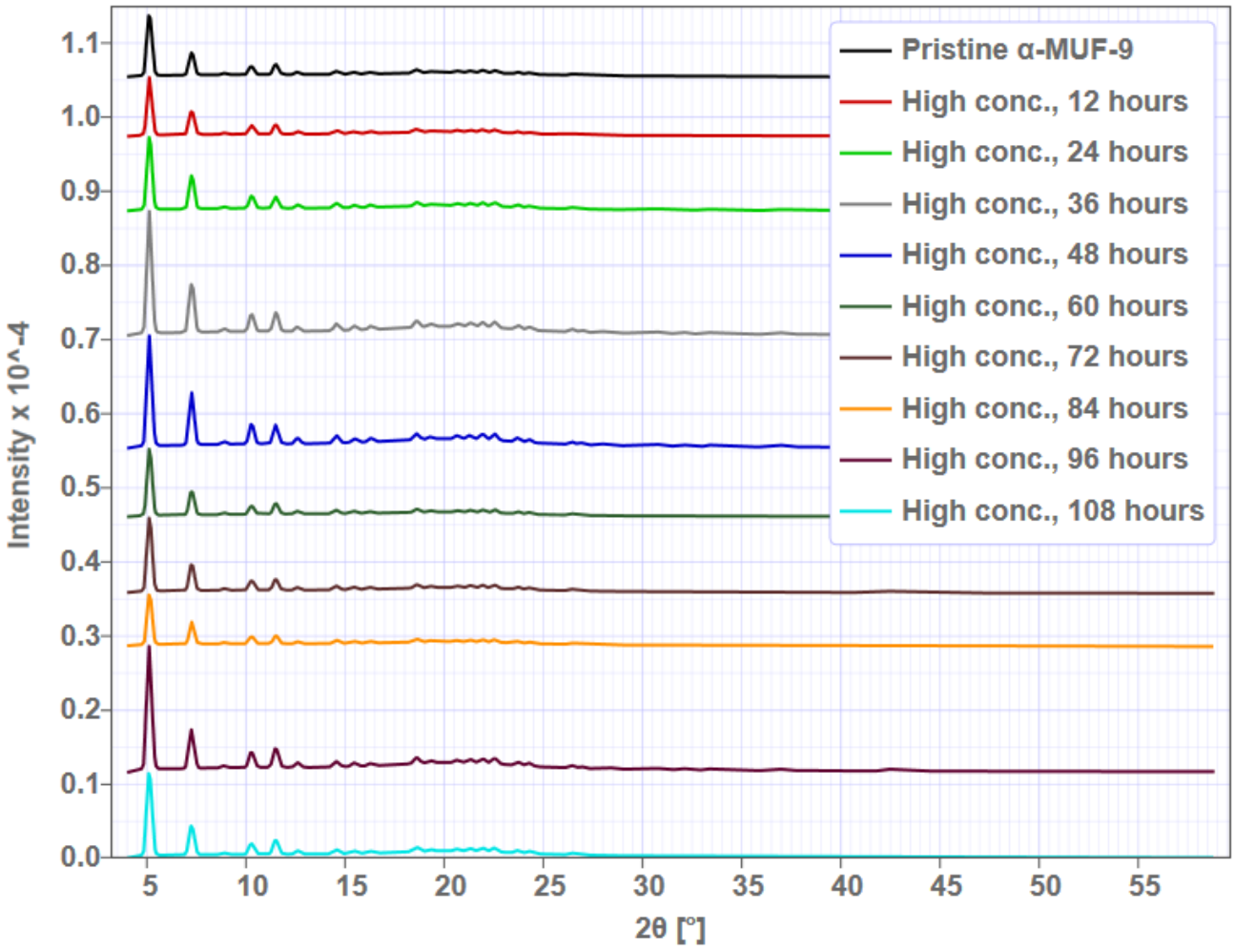


Figure S25: PXRD patterns α-MUF-9 heated with a 20 mg mL^{-1} solution of $Co(NO_3)_2 \cdot 6H_2O$ in DBF at 95 °C.

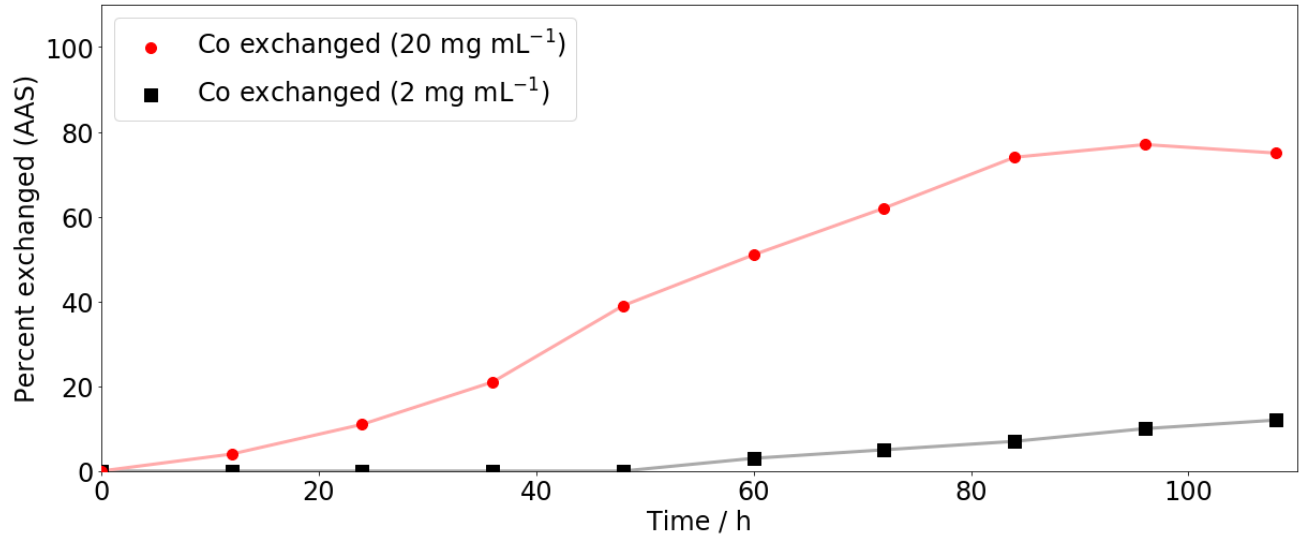


Figure S26: Plot of percentages of zinc(II) displaced by cobalt(II) in α-MUF-9 over time as determined by AAS. Black squares represent percentage exchange with the 2 mg mL^{-1} cobalt nitrate solution, while the red circles represent exchange with the 20 mg mL^{-1} cobalt nitrate solution.

S5.3. Heating α-MUF-9 with biphenyldicarboxylate ligands

Stock solutions of 0.5 mg mL^{-1} H_2bpdc or 1 mg mL^{-1} H_2bpdc-NH_2 with 3 mg mL^{-1} 2-fluorobenzoic acid were prepared in DBF and added to vials of α-MUF-9 crystals. The vials were placed in a dry bath set to 85 °C and the stock solution replaced every 6 hours. PXRD patterns were obtained from each sample then the samples washed with DMF (2 mL × 3) and acetone (4 mL × 5) and dried under high vacuum. Each sample was then digested in 0.2 M NaOD in D_2O for ^{1}H NMR spectroscopy.

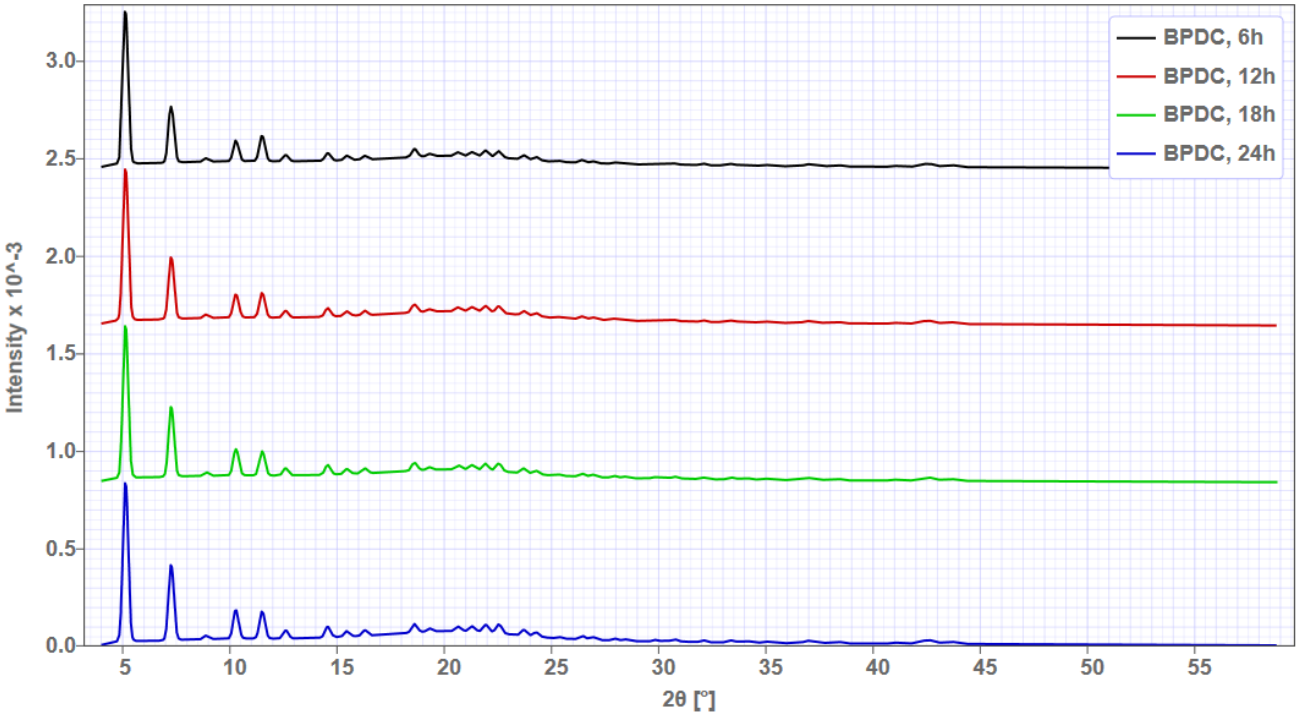


Figure S27: PXRD patterns of α-MUF-9 heated in a 0.5 mg mL^{-1} H_2bpdc solution in DBF.

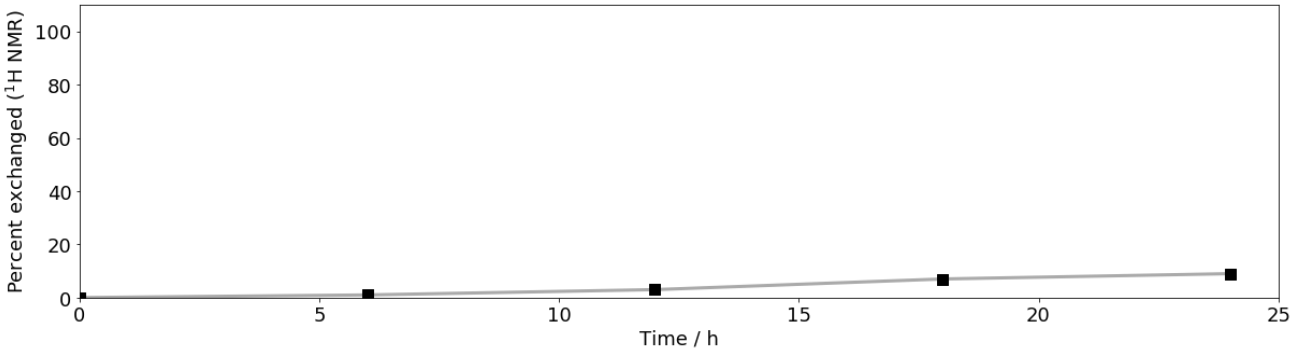


Figure S28: The amount of **L1** displaced by bpdc in α-MUF-9 over time as determined by ^{1}H NMR spectroscopy of digested samples.

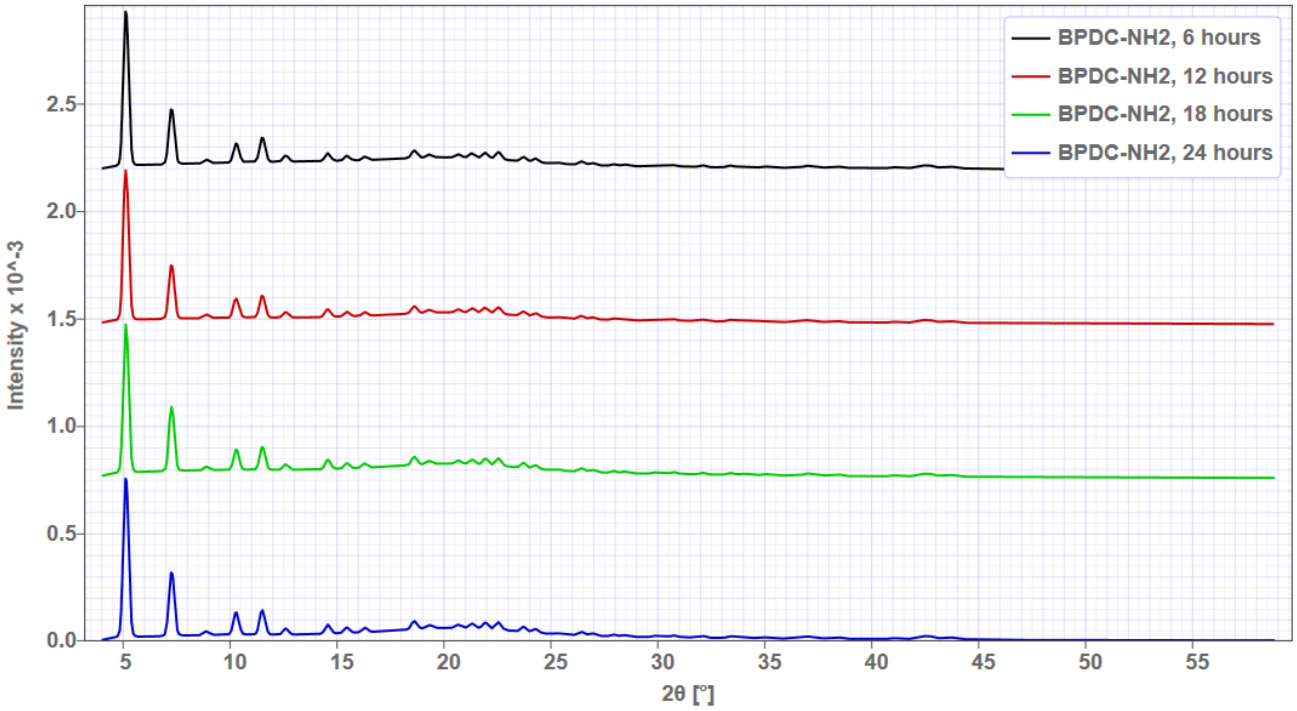


Figure S29: PXRD patterns of α-MUF-9 heated with a 1 mg mL^{-1} H_2bpdc-NH_2 solution in DBF.

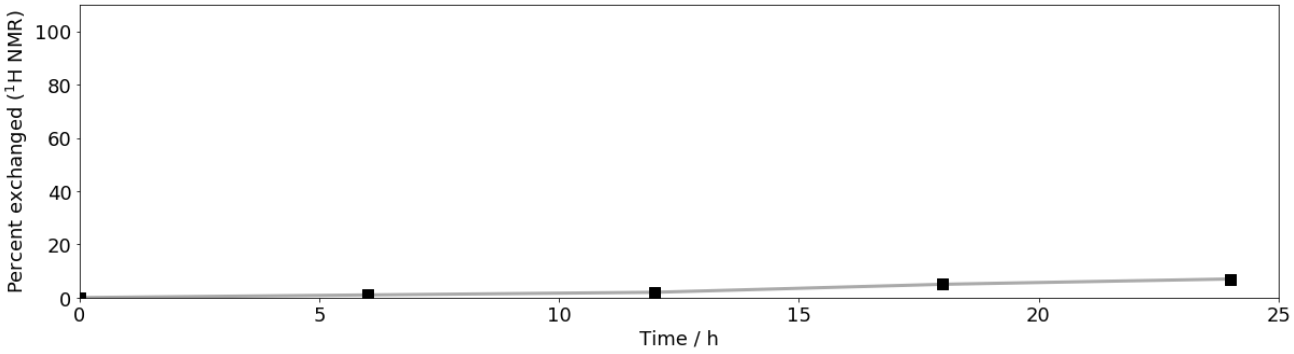


Figure S30: The amount of **L1** displaced by bpdc-NH_2 in α-MUF-9 over time as determined by ^{1}H NMR spectroscopy of digested samples.

S6. Python scripts

Python codes are provided here for review and the convenience of anyone wanting to replicate or improve upon this work. They have no license attached and may be used, adapted, modified, or redistributed for any purpose, commercial or not, without attribution. These codes are provided "as is", without warranty of any kind, express or implied, including, fitness for a particular purpose and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, arising from use of the software.

S6.1. Correction of baselines for PXRD patterns

This code is in two parts. PXRD_process.py handles the files. It runs on all Rigaku .asc format files in a folder named ‘input’ and writes baseline corrected .xy files to a folder named output. It uses the algorithm in sonneveld.py as a dependency.

sonneveld.py

```
#Modified Sonneveld-Visser Baseline Removal Algorithm
#inputs is a single numpy 2Darray containting xy format PXRD data
#Sonneveld algorithm selects baseline points by choosing ones where the gradient to the next point is small
#then a polynomial fit to those points is subtracted from the pattern
#as described in J. Appl. Crystallogr. (1975), 8, p1
import numpy as np
import random

def remove baseline(inputs,step,iter,maxpeak,poly_order):
        input = inputs[1]
        maximum = max(input)
        print(maximum)
        points = list(range(int(len(input)*0.7)))[1::step]
        for j in range(iter):
                change = False
                for i in range(len(points)-1):
                        if (abs(input[points[i]]-input[points[i+1]]) > (maxpeak * maximum)):
                                print("large change detected at " + str(i))
                                change = True
                                for k in range(i,len(points)):
                                        points[k] += step
                                        if points[k] >= len(input): points[k] =
random.choice(range(int(len(input)*0.7)))
                if not change: break
                print("reiterating, attempt " + str(j))
        x = list([inputs[0][i] for i in points])
        y = list([inputs[1][i] for i in points])
        fit = np.polyfit(x,y,poly_order)
        print(fit)
        f = np.poly1d(fit)
        outputdata = [(inputs[1][i] - f(inputs[0][i])) for i in range(len(inputs[0]))]
        outputdata = [max(i,0) for i in outputdata]

        return [inputs[0],outputdata]
```

PXRD_process.py

```
import os
import re
import sonneveld
import numpy as np
#import all files from /input folder
filelist = os.listdir("./input/")
print(filelist)

def convert_to_xy(f):
        start = 0
        stop = 0
        step = 0
        count = 0
        data = []
        for i in range(len(f)):
                if "*START" in f[i]: start = float(re.findall("\d+\.\d+", f[i])[0])
                if "*STOP" in f[i]: stop = float(re.findall("\d+\.\d+", f[i])[0])
                if "*STEP" in f[i]: step = float(re.findall("\d+\.\d+", f[i])[0])

                if "*COUNT " in f[i]:
                        count = int(re.findall("\d+", f[i])[0])
                        for j in [(x + i + 1) for x in range (count)]:
                                data.append(float(f[j]))
        points = ([round((start+i*step),2) for i in range(int((stop - start)/step))])
        return [points,data]

def scale(input,endmax):
        firstmax = max(input[1])
        scalefactor = endmax/firstmax
        out=[x * scalefactor for x in input[1]]
        return [input[0],out]

#main routine
for i in range(len(filelist)):
        print("processing "+filelist[i])
        ftext = open("./input/"+filelist[i], 'r').read().splitlines()
        xyf = convert_to_xy(ftext)
        xyf[1].pop()
        print(len(xyf[0]))
        print(len(xyf[1]))
        xyf = sonneveld.remove_baseline(xyf,10,200,0.05,10)
        xyf = scale(xyf,1000)
        outname = filelist[i][:-4] + ".txt"
        otext = open("./output/"+outname, 'w')
        for j in range(len(xyf[0])):
                print(str(xyf[0][j]) + " " + str(xyf[1][j]), file=otext)
```

S6.2. Interpenetration percentage determination

Takes a folder of sample data in .xy format, a fully interpenetrated PXRD pattern, and a noninterpenetrated PXRD pattern as command line arguments. Returns as console output the percentage contribution of the interpenetrated pattern for each sample in the input folder, as determined by the linear mixture of interpenetrated and noninterpenetrated patterns with the least squared-differences to the sample pattern.

```
import numpy as np
import sys
import os

folder = "./"+sys.argv[1]+"/"
samples = os.listdir(folder)

def scale(input,endmax):
        firstmax = max(input)
        scalefactor = endmax/firstmax
        return [x * scalefactor for x in input]


def importArray(filename):
        ftext = open(filename, 'r').read().splitlines()
        list1 = []
        for x in ftext:
                tmp = x.split()
                cur = float(tmp[0])
                #RANGE OF 2-THETA TO USE - for MUF-9 this works best with the two first peaks
                if 4.7 < cur < 8:
                        list1.append(float(tmp[1]))
        return np.array(scale(list1,1000))

def mix(array1,array2,contribution):
        return np.add(np.multiply(array1,1-contribution),np.multiply(array2,contribution))

def getSSD(reference,sample):
        sample_scaled = sample * (reference.max() / sample.max())
        squares = (reference - sample_scaled) ** 2
        return np.sum(squares)

nonint_array = importArray(sys.argv[2])
int_array = importArray(sys.argv[3])

def refine(samplename):
        test_array = importArray(folder + samplename)
        contribution = 0.0
        step = 0.01
        improvement = True
        ssd = getSSD(mix(nonint_array,int_array,contribution),test_array)

        while improvement:
                contribution += step
                old_ssd = ssd
                ssd = getSSD(mix(nonint_array,int_array,contribution),test_array)
                if ssd > old_ssd:
                        improvement = False

        return contribution

for sample in samples:
        print(sample, refine(sample))
```

S6.3. Generation of difference data

Must be run in a cctbx.python environment. Takes the high-energy and metal absorption edge datasets as first and second command line arguments respectively.

```
from iotbx.reflection_file_reader import any_reflection_file
from cctbx import miller
from cctbx import crystal
from cctbx.array_family import flex
from iotbx.command_line import patterson_map
import sys

#Cell and symmetry data for the datasets - no way to obtain these from SHELX .hkl files, but could be
extracted from other filetypes
symm = crystal.symmetry(
       space_group_symbol="P-43m",
       unit_cell=(17.1,17.1,17.1,90,90,90))

print "Reading reflection files, merging equivalents \n\n\n"
#read and format high-energy dataset
hkl_in_native = any_reflection_file(file_name=sys.argv[1]+"=hklf4")
miller_set_native =
hkl_in_native.as_miller_arrays()[0].customized_copy(crystal_symmetry=symm,anomalous_flag=False).merge_equival
ents().array()
miller_set_native.show_summary()

print "\n"
#read and format high-energy dataset
hkl_in_derivative = any_reflection_file(file_name=sys.argv[2]+"=hklf4")
miller_set_derivative = hkl_in_
derivative.as_miller_arrays()[0].customized_copy(crystal_symmetry=symm,anomalous_flag=False).merge_equivalent
s().array()
miller_set_derivative.show_summary()

print "\n\n Scaling reflection files and finding common reflections \n\n"
#Use only the reflections available in the derivative (lower resolution) dataset
miller_set_native, miller_set_derivative = miller_set_native.common_sets(other=miller_set_ derivative)
miller_set_derivative = miller_set_native.multiscale(other=miller_set_derivative)
delta_f = miller_set_native.customized_copy(data=miller_set_native.data()-miller_set_derivative.data())

#print some stats for sanity checking
miller_set_native.show_summary()
print list(miller_set_native.data())[0:20]
print "\n"
miller_set_derivative.show_summary()
print list(miller_set_derivative.data().as_float())[0:20]

#output SHELX-format hkl file, and CCP4-format patterson map
print "\n writing difference hkl file \n"
f = open("delta_f.hkl", "w")
delta_f.export_as_shelx_hklf(file_object=f)
f.close()
pmap = patterson_map.calculate_patterson_map(data=delta_f, params=params)
pmap.as_ccp4_map(file_name="delta_f_patterson")
```

S6.4. Integration of difference data

Generates various information and statistics from Gaussian cube format electron density maps.

This depends on the *cubetools* library written by P. R. Vaidyanathan and published at https://gist.github.com/aditya95sriram/8d1fccbb91dae93c4edf31cd6a22510f under the MIT license. This library is appended below for convenience.

Atom locations and sizes are obtained from the refinement of signals in difference datasets (S4.1.3) and hardcoded into this script, which counts the total peak intensity over the volume of a sphere at each location in the cube map, the maximum peak height at each atom site, and the noise level as root mean squared deviation of each map,and returns these as console output.

electron_density_integration.py

```
import numpy as np
import math
import cubetools
from scipy.spatial import distance

def is_in_sphere(origin, radius, point):
    if (distance.euclidean(origin, point) <= radius):
        #print (origin, radius, point)
        #print ("distance ", distance.euclidean(origin, point), "=> true")
        return True
    else:
        #print (origin, radius, point)
        #print ("distance ", distance.euclidean(origin, point), "=> false")
        return False

def integrate_atom(location, radius, cell_size, emap):
    #location should be a triple (x,y,z) in fractional coordinates,
    #radius and cell_size should be in Angstroms
    sites = []
    min_i = int(( location[0] - (radius / cell_size) ) * emap.shape[0])
    max_i = int(( location[0] + (radius / cell_size) ) * emap.shape[0])
    min_j = int(( location[1] - (radius / cell_size) ) * emap.shape[1])
    max_j = int(( location[1] + (radius / cell_size) ) * emap.shape[1])
    min_k = int(( location[2] - (radius / cell_size) ) * emap.shape[2])
    max_k = int(( location[2] + (radius / cell_size) ) * emap.shape[2])
    print("integrating from x = "+str(min_i)+" to "+str(max_i)+", y = "+str(min_j)+" to "+str(max_j)+", z =
"+str(min_k)+" to "+str(max_k))
    for i in range(min_i, max_i):
        for j in range(min_j, max_j):
            for k in range(min_k, max_k):
                x = (i/emap.shape[0])
                y = (j/emap.shape[1])
                z = (k/emap.shape[2])
                if is_in_sphere(location, radius/cell_size, (x,y,z)): sites.append(emap[i,j,k])
    return sites

fo_emap = cubetools.read_cube("MUF-93-72h-fobs.cube")
fo_emap_array = fo_emap[0]
fo_min_fc_emap = cubetools.read_cube("MUF-93-72h-fobs-min-fcalc.cube")
fo_min_fc_emap_array = fo_min_fc_emap[0]

atom_1 = (integrate_atom((INSERT ATOM COORDINATE HERE), 1.7, 17.1, fo_emap_array))
atom_2 = (integrate_atom((INSERT ATOM COORDINATE HERE), 1.7, 17.1, fo_emap_array))
print (atom_1, atom_2)
print(sum(atom_1), sum(atom_2))

print(fo_emap_array.shape)
print(fo_emap_array)
print(fo_min_fc_emap_array.shape)
print(fo_min_fc_emap_array)

def map_rmsd(emap_arr):
    emap_mean = emap_arr.mean(axis=None)
    print ("mean value:",emap_mean,"min value:",emap_arr.min(),"max value:",emap_arr.max())
    emap_disp = emap_arr - emap_mean
    emap_disp_sq = np.square(emap_disp)
    emap_mean_disp_sq = emap_disp_sq.mean()
    print ("mean squared displacement: " + str(emap_mean_disp_sq))
    emap_rmsd = math.sqrt(emap_mean_disp_sq)
    print ("root mean squared displacement: " + str(emap_rmsd))
```

```
    return emap_rmsd

print("Statistics for (Fobs, phi) electron density map:")
noise_fobs = map_rmsd(fo_emap_array)
print("")
print("Statistics for (Fobs - Fcalc, phi) electron density map:")
noise_diff = map_rmsd(fo_min_fc_emap_array)
print("")
print("meax peak height at location 1:", max(atom_1))
print("meax peak height at location 2:", max(atom_2))
print("integrated electron density at locations 1 & 2:", sum(atom_1), sum(atom_2))
print("ratio of peak integrals:", sum(atom_2)/sum(atom_1)*100, "%" )
print("")
print("signal to noise ratio for main peak (noise calculated from (Fobs, phi) = ", max(atom_1)/noise_fobs)
print("signal to noise ratio for main peak (noise calculated from (Fobs - Fcalc, phi)) = ",
max(atom_1)/noise_diff)
print("")
```

cubetools.py

```
#-------------------------------------------------------------------------------
# Module: cubetools
#-------------------------------------------------------------------------------
#
# Description:
# Module to work with Gaussian cube format files
# (see http://paulbourke.net/dataformats/cube/)
#
#-------------------------------------------------------------------------------
#
# What does it do:
# * Read/write cube files to/from numpy arrays (dtype=float*)
# * Read/write pairse of cube files to/from numpy arrays (dtype=complex*)
# * Provides a CubeFile object, to be used when cubefiles with
#   constant and static data is required. It simulates the readline method
#   of a file object with a cube file opened, without creating a file
#
#-------------------------------------------------------------------------------
#
# Dependency: numpy
#
#-------------------------------------------------------------------------------
#
# Author: P. R. Vaidyanathan (aditya95sriram <at> gmail <dot> com)
# Date: 25th June 2017
#
#-------------------------------------------------------------------------------
#
# MIT License
#
# Copyright (c) 2019 P. R. Vaidyanathan
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
#
#-------------------------------------------------------------------------------

import numpy as np

if __name__ == '__main__':
    DEBUGMODE = True
else:
    DEBUGMODE = False

def _debug(*args):
    global DEBUGMODE
    if DEBUGMODE:
        print " ".join(map(str, args))

class CubeFile(object):
    """
    Object which mimics a cube file opened as a file object
    by returning output in the correct format, matching the
    metadata of the source cube file and replacing volumetric
    data with static data provided as arg to the constructor.
    Doesn't copy atoms metadata, retains number of atoms, but
    returns dummy atoms
    Mimics file object's readline method.

    params:
        srcname: source file to copy metadata from
        const: numeric value to return instead of volumetric data

    returns: CubeFile object
```

```
    """

    def __init__(self, srcname, const=1):
        self.cursor = 0
        self.const = const
        self.src = src = open(srcname)
        src.readline(); src.readline(); # comments
        _debug(srcname)
        self.lines = [" Cubefile created by cubetools.py\n",
                      "  source: {0}\n".format(srcname)]
        self.lines.append(src.readline()) # read natm and origin
        self.natm = int(self.lines[-1].strip().split()[0])
        # read cube dim and vectors along 3 axes
        self.lines.extend(src.readline() for i in range(3))
        self.src.close()
        self.nx, self.ny, self.nz = [int(l.strip().split()[0]) for l in self.lines[3:6]]
        self.remvals = self.nz
        self.remrows = self.nx*self.ny
        for i in range(self.natm):
            self.lines.append("{0:^ 8d}".format(1) + "{0:< 12.6f}".format(0)*4 + '\n')

    def __del__(self):
        self.src.close()

    def readline(self):
        """ Mimic readline method of file object with cube file opened """
        try:
            retval = self.lines[self.cursor]
        except IndexError:
            if not self.remrows:
                return ""
            if self.remvals <= 6:
                nval = min(6,self.remvals)
                self.remrows -= 1
                self.remvals = self.nz
            else:
                nval = 6
                self.remvals -= nval
            return " {0: .5E}".format(self.const)*nval + "\n"
        else:
            self.cursor += 1
            return retval

def  getline(cube):
    """
    Read a line from cube file where first field is an int
    and the remaining fields are floats.

    params:
        cube: file object of the cube file

    returns: (int, list<float>)
    """
    l = cube.readline().strip().split()
    return int(l[0]), map(float, l[1:])

def _putline(*args):
    """
    Generate a line to be written to a cube file where
    the first field is an int and the remaining fields are floats.

    params:
        *args: first arg is formatted as int and remaining as floats

    returns: formatted string to be written to file with trailing newline
    """
    s = "{0:^ 8d}".format(args[0])
    s += "".join("{0:< 12.6f}".format(arg) for arg in args[1:])
    return s + "\n"

def read_cube(fname):
    """
    Read cube file into numpy array

    params:
        fname: filename of cube file

    returns: (data: np.array, metadata: dict)
    """
    meta = {}
    with open(fname, 'r') as cube:
        cube.readline(); cube.readline()  # ignore comments
```

```
        natm, meta['org'] = _getline(cube)
        nx, meta['xvec'] = _getline(cube)
        ny, meta['yvec'] = _getline(cube)
        nz, meta['zvec'] = _getline(cube)
        meta['atoms'] = [_getline(cube) for i in range(natm)]
        data = np.zeros((nx*ny*nz))
        idx = 0
        for line in cube:
            for val in line.strip().split():
                data[idx] = float(val)
                idx += 1
    data = np.reshape(data, (nx, ny, nz))
    return data, meta

def read_imcube(rfname, ifname = ""):
    """
    Convenience function to read in two cube files at once,
    where one contains the real part and the other contains the
    imag part. If only one filename given, other filename is inferred.

    params:
        rfname: filename of cube file of real part
        ifname: optional, filename of cube file of imag part

    returns: np.array (real part + j*imag part)
    """
    ifname = ifname or rfname.replace('real', 'imag')
    _debug("reading from files", rfname, "and", ifname)
    re, im = read_cube(rfname), read_cube(ifname)
    fin = np.zeros(re[0].shape, dtype='complex128')
    if re[1] != im[1]:
         debug("warning: meta data mismatch, real part metadata retained")
    fin += re[0]
    fin += 1j*im[0]
    return fin, re[1]

def write_cube(data, meta, fname):
    """
    Write volumetric data to cube file along

    params:
        data: volumetric data consisting real values
        meta: dict containing metadata with following keys
            atoms: list of atoms in the form (mass, [position])
            org: origin
            xvec,yvec,zvec: lattice vector basis
        fname: filename of cubefile (existing files overwritten)

    returns: None
    """
    with open(fname, "w") as cube:
        # first two lines are comments
        cube.write(" Cubefile created by cubetools.py\n  source: none\n")
        natm = len(meta['atoms'])
        nx, ny, nz = data.shape
        cube.write(_putline(natm, *meta['org'])) # 3rd line #atoms and origin
        cube.write(_putline(nx, *meta['xvec']))
        cube.write(_putline(ny, *meta['yvec']))
        cube.write(_putline(nz, *meta['zvec']))
        for atom_mass, atom_pos in meta['atoms']:
            cube.write(_putline(atom_mass, *atom_pos)) #skip the newline
        for i in range(nx):
            for j in range(ny):
                for k in range(nz):
                    if (i or j or k) and k%6==0:
                        cube.write("\n")
                    cube.write(" {0: .5E}".format(data[i,j,k]))

def write_imcube(data, meta, rfname, ifname=""):
    """
    Convenience function to write two cube files from complex valued
    volumetric data, one for the real part and one for the imaginary part.
    Data about atoms, origin and lattice vectors are kept same for both.
    If only one filename given, other filename is inferred.

    params:
        data: volumetric data consisting complex values
        meta: dict containing metadata with following keys
            atoms: list of atoms in the form (mass, [position])
            org: origin
            xvec,yvec,zvec: lattice vector basis
        rfname: filename of cube file containing real part
```

```
    ifname: optional, filename of cube file containing imag part

returns: None
"""
ifname = ifname or rfname.replace('real', 'imag')
_debug("writing data to files", rfname, "and", ifname)
write_cube(data.real, meta, rfname)
write_cube(data.imag, meta, ifname)
```

S7. References

1. Ferguson, A.; Liu, L.; Tapperwijn, S. J.; Perl, D.; Coudert, F.-X.; Van Cleuvenbergen, S.; Verbiest, T.; van der Veen, M. A.; Telfer, S. G., Controlled partial interpenetration in metal–organic frameworks. *Nat. Chem.* **2016,** *8*, 250-257.
2. Allinger, N. L.; Youngdale, G. A., Reduction of 6,7-Diphenyldibenzo[e,g] [1,4]diazocine. An Unusual Nucleophilic Aromatic Substitution 1. *J. Org. Chem.* **1959,** *24*, 306-308.
3. Hu, Y. H.; Wang, J. C.; Yang, S.; Li, Y. A.; Dong, Y. B., CuI@UiO-67-IM: A MOF-Based Bifunctional Composite Triphase-Transfer Catalyst for Sequential One-Pot Azide-Alkyne Cycloaddition in Water. *Inorg. Chem.* **2017,** *56*, 8341-8347.
4. Cowieson, N. P.; Aragao, D.; Clift, M.; Ericsson, D. J.; Gee, C.; Harrop, S. J.; Mudie, N.; Panjikar, S.; Price, J. R.; Riboldi-Tunnicliffe, A.; Williamson, R.; Caradoc-Davies, T., MX1: a bending-magnet crystallography beamline serving both chemical and macromolecular crystallography communities at the Australian Synchrotron. *J Synchrotron Radiat* **2015,** *22*, 187-90.
5. Aragão, D.; Aishima, J.; Cherukuvada, H.; Clarken, R.; Clift, M.; Cowieson, N. P.; Ericsson, D. J.; Gee, C. L.; Macedo, S.; Mudie, N., MX2: a high-flux undulator microfocus beamline serving both the chemical and macromolecular crystallography communities at the Australian Synchrotron. *Journal of synchrotron radiation* **2018,** *25*, 885-891.
6. Sheldrick, G., SHELXT - Integrated space-group and crystal-structure determination. *Acta Crystallographica Section A* **2015,** *71*, 3-8.
7. Sheldrick, G., Crystal structure refinement with SHELXL. *Acta Crystallogr. C* **2015,** *71*, 3-8.
8. Dolomanov, O. V.; Bourhis, L. J.; Gildea, R. J.; Howard, J. A. K.; Puschmann, H., OLEX2: a complete structure solution, refinement and analysis program. *J. Appl. Crystallogr.* **2009,** *42*, 339-341.
9. Kabsch, W., Xds. *Acta Crystallogr. D* **2010,** *66*, 125-32.
10. Grosse-Kunstleve, R. W.; Sauter, N. K.; Moriarty, N. W.; Adams, P. D., The Computational Crystallography Toolbox: crystallographic algorithms in a reusable software framework. *J. Appl. Crystallogr.* **2002,** *35*, 126-136.